



Some results from the CALIBRE project

Olivier BERGER (GET/INT – France)

ObjectWeb and OrientWare Open Source
software seminar

CISIS 2006

Dalian (China), 23 June 2006



This work is licensed under a Creative Commons Attribution–ShareAlike License.
<http://creativecommons.org/licenses/by-sa/2.0/>

Contents

- Intro
- Calibre's context
- Some results
 - software engineering
 - sociology, economics
 - dissemination
- New characterisation of OSS ?

About GET & INT

- GET is a group of several public higher education schools in France :
 - teaching + research
 - field of Telecommunication and IT
- In GET, INT (National Institute of Telecommunications), near Paris: business school + engineering school
- Several teams specialised in research and practice on Libre Software



General context

Definition of Free/Libre/Open Source software

- ***CALIBRE*** : « *libre software* »
- « Libre », as in liberty (or free as in freedom)
- [Free Software / Open Source] licence
- Several names, same phenomenon
- Free + Libre + OSS = FLOSS ...



Translations of *Libre* software

- Chinese : 自由軟件 / 自由軟體 (?)
- Japanese: 自由なソフトウェア (?)
(フリ ソフトウェア) (?)
- Korean: 자유 소프트웨어 (?)
- French: logiciel ***libre***
- Italian: software ***libero***
- Spanish: software ***libre***



CALIBRE project



Context of CALIBRE project

- European Community (EC)
- DG Information Society of European Commission
- 6th Framework Programme (FP 6) : R&D funding programme of EC
- Academic consortium : research by academic institutions funded in FP6
- FP6 ending in 2006 (FP7)



« Coordination Action for LIBRE software »

- IST FP6 Project : 2 year : 2004 to 2006
- Ending july (september ?) 2006
- Multi-disciplinary research team :
 - Economy,
 - Software Engineering,
 - Sociology, ...
- Critical mass of Europe's academic research in Libre software

CALIBRE Partners

- Universities and research centers in 12 European countries + China
 - In France : GET + UPMC
- More details on Calibre on <http://www.calibre.ie/>





Goals of CALIBRE

FLOSS as a 'silver bullet'

- Proponents claim FLOSS can solve "software crisis" (cost, quality and duration of development)
- Research needed to confirm
- Not one only model
- Future model for work and society
 - Wikipedia, open science, human genome
- Pitfalls ?
 - FLOSS and Navajo Indians!

Why EC funds research on FLOSS (> 1.5 M euros)

- Libre/Open Source software model seen as big potential for European Industry
- To the next generation methods and services ?
- From FLOSS to OSS 2.0 ?
- Foster Academic research / clustering
- Transfer lessons to the industry
(Calibration industry forum)



Software Engineering

Software engineering challenges

- Huge amount of freely available public data relating to libre software development projects
- Successful development model(s)
- Hope that data obtained from public sources can help understand the undergoing processes



Concepts and techniques

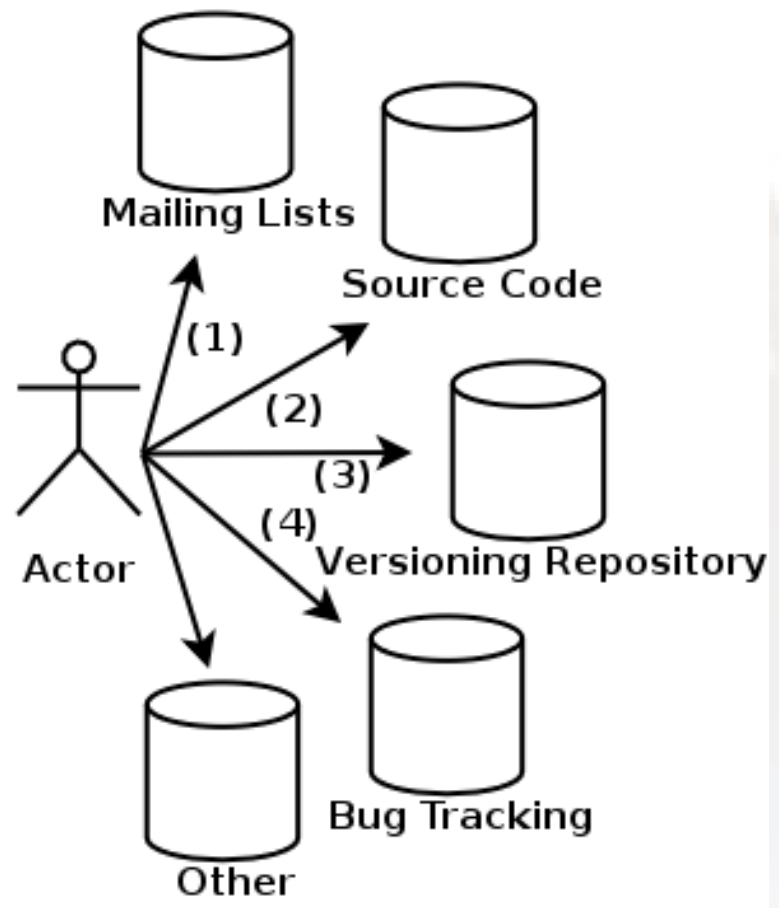
- « Cathedral and bazaar » (Eric S. Raymond) 1997
 - « Cathedral » : heavyweight process in hierarchical structure
 - « Bazaar » : loosely coordinated development teams
 - Libre software community's own research
- Academic researchers have become interested for several years



Building a discipline

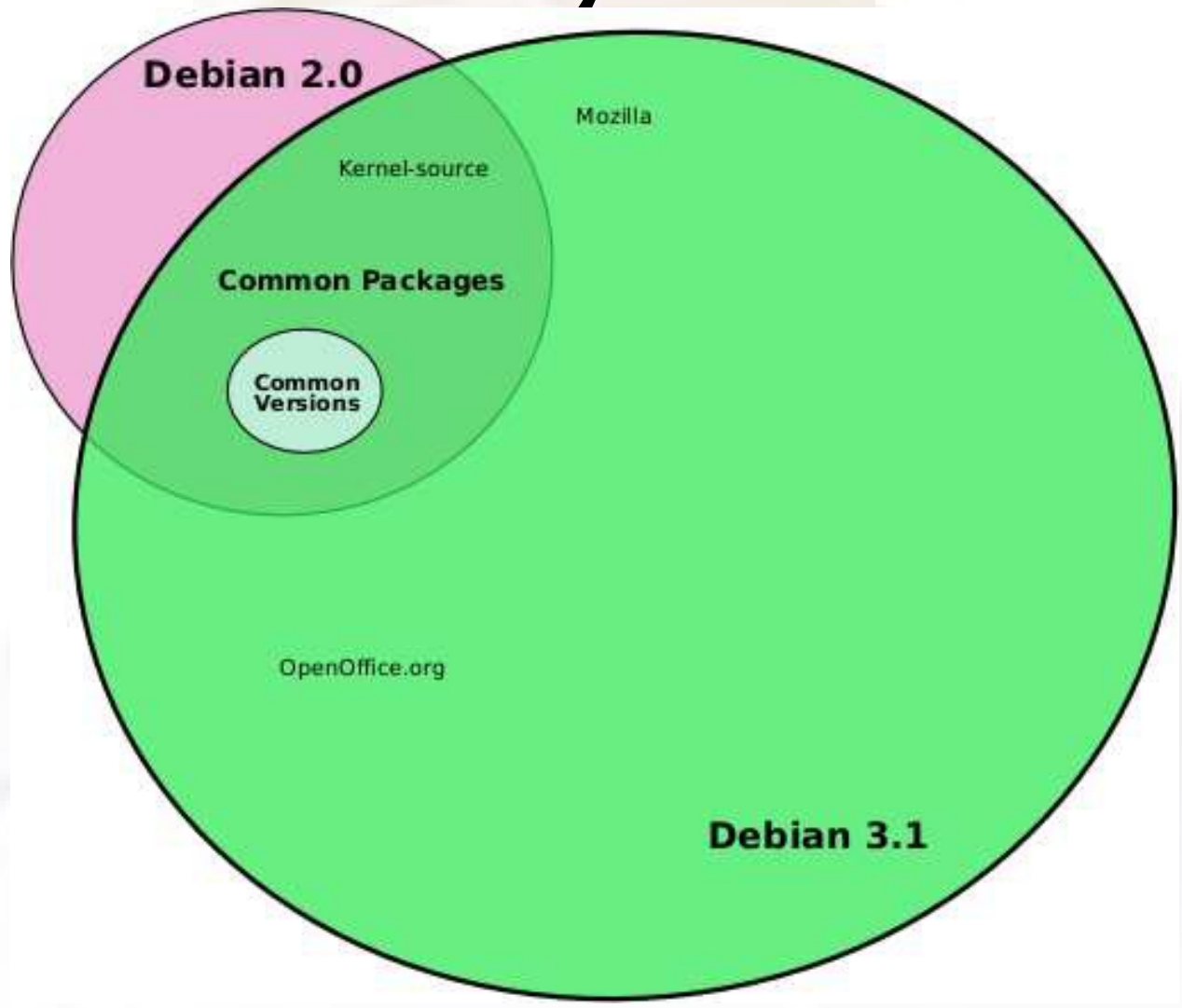
- Browsing source code to identify authors and metrics
- Some research paths :
 - Research in revisions repositories
 - Social networks analysis
 - Software evolution
- Tools to automate development repositories mining

Public data sources





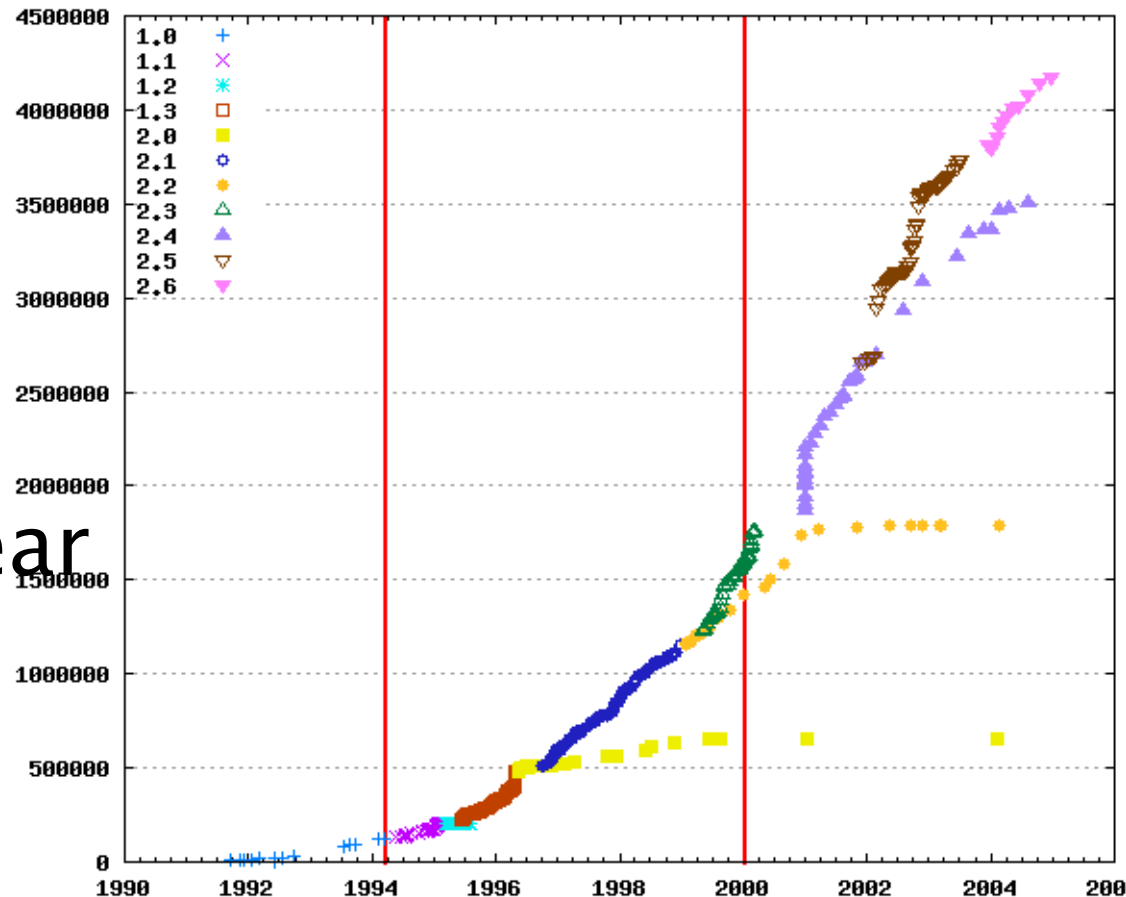
Macro analysis : Distributions



Source
[Robles]

Evolution of one software (SLOC grow)

- "Classical" methodology
- Usual profile : linear
- Linux : superlinear



Linux kernel source line count evolution Source: [Robles]

(BETA3) CVS Analysis for the KDE project

Contents

- [Index](#)
- [About](#)
- [Statistics](#)
- [Modules](#)
- [Committers](#)
- [Inequality](#)
- [FAQ](#)
- [Credits](#)

Module Search

Go

Committer Search

Go

Language

English

Go

(BETA3) CVS Analysis for the KDE project - General Statistics

Historical data

First commit	1997-04-09 00:25:19
Last commit considered (*)	2004-03-22 20:59:43
Number of days	2539.9

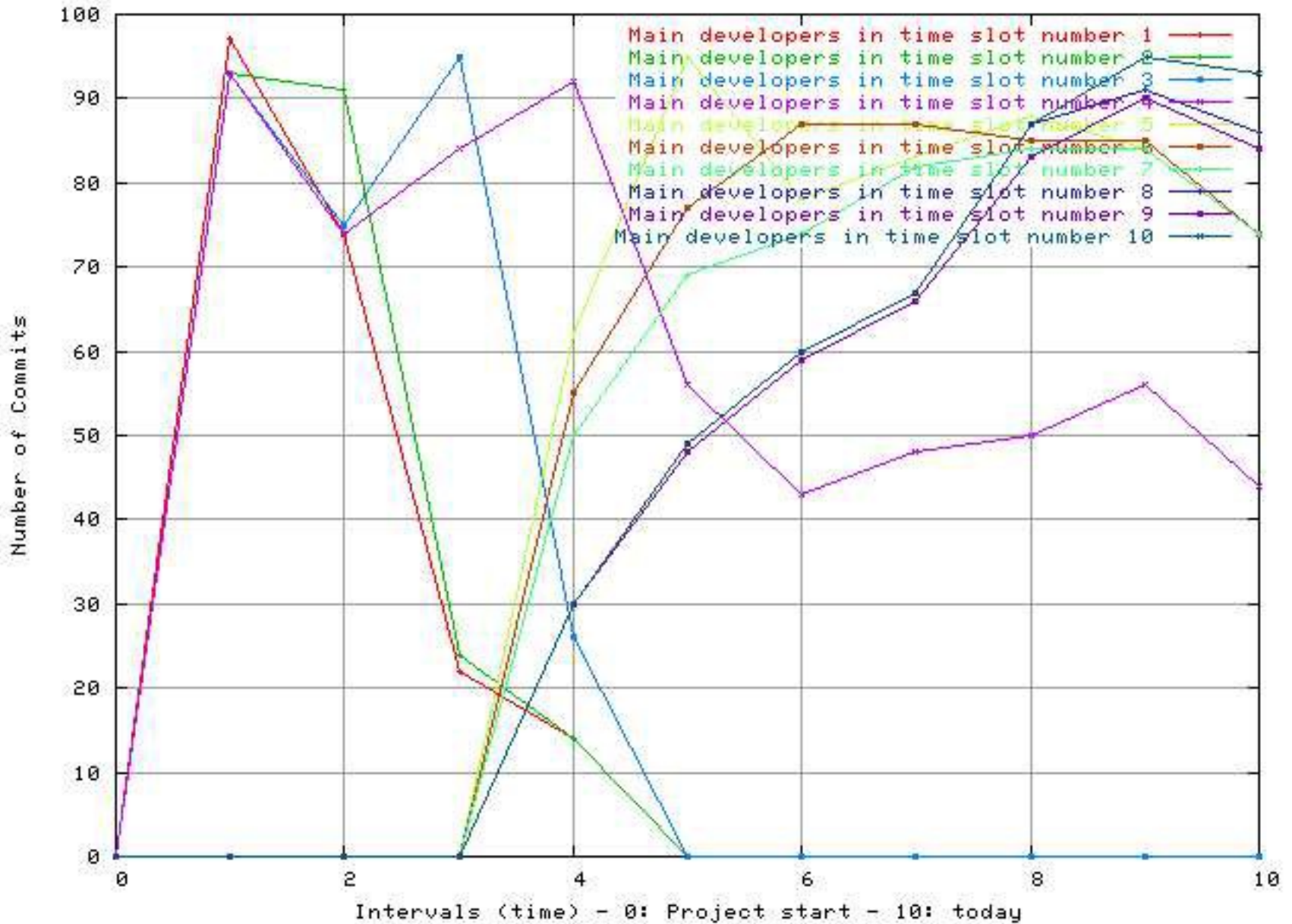
(*) CVSAnalY analysis date. This date is considered as the reference point for further analysis.

	Number	Mean per module	Mean per commiter	Mean per commit	Mean per day
Modules	79	1	0.09	3E-05	0.0311
Committers	915	11.58	1	0.0003	0.36
Commits	2935436	37157.42	3208.13	1	1155.73
Files	175657	2223.51	191.97	0.06	69.16
Aggregated Lines	106048029	1342380.11	115899.49	36.13	41752.84
Removed Lines	73534466	930816.03	80365.54	25.05	28951.72
Changed lines	179582495	2273196.14	196265.02	61.18	70704.55
Final Lines	32513563	411564.09	35533.95	11.08	12801.12

File-type statistics for all modules

File type	Modules	Commits	Files	Lines Changed	Lines Added	Lines Removed	Removed External files	CVS flag	First commit	Last commit
development	74	1061173	76505	36989453	25107823	11881630	87738	9428	101860	1997-04-10 2004-03-22
i18n	26	815279	2305	114107944	61994713	52113231	64926	96	724323	1997-08-15 2004-03-22
documentation	64	461647	24242	16104796	10840701	5264095	85235	151	346241	1997-04-13 2004-03-22

Evolution of commits in time for top committers by percentage for each time interval



Source : [CVSAnalY]

Developpers



Counting in SourceForge

- **By countries:**

Rank	Country	Developers
1.	United States	425620
2.	Germany	95800
3.	United Kingdom	60768
4.	Canada	49109
5.	France	44587
6.	China	36517
...





Counting in SourceForge (2)

- **By regions:**

Region	Developers
Africa	12560
Asia	127275
EU	401845
Europe	466792
North America	485679
Oceania	46422
South America	36330

(source : Gregorio Robles and Jesús M. González Barahona – 2006)

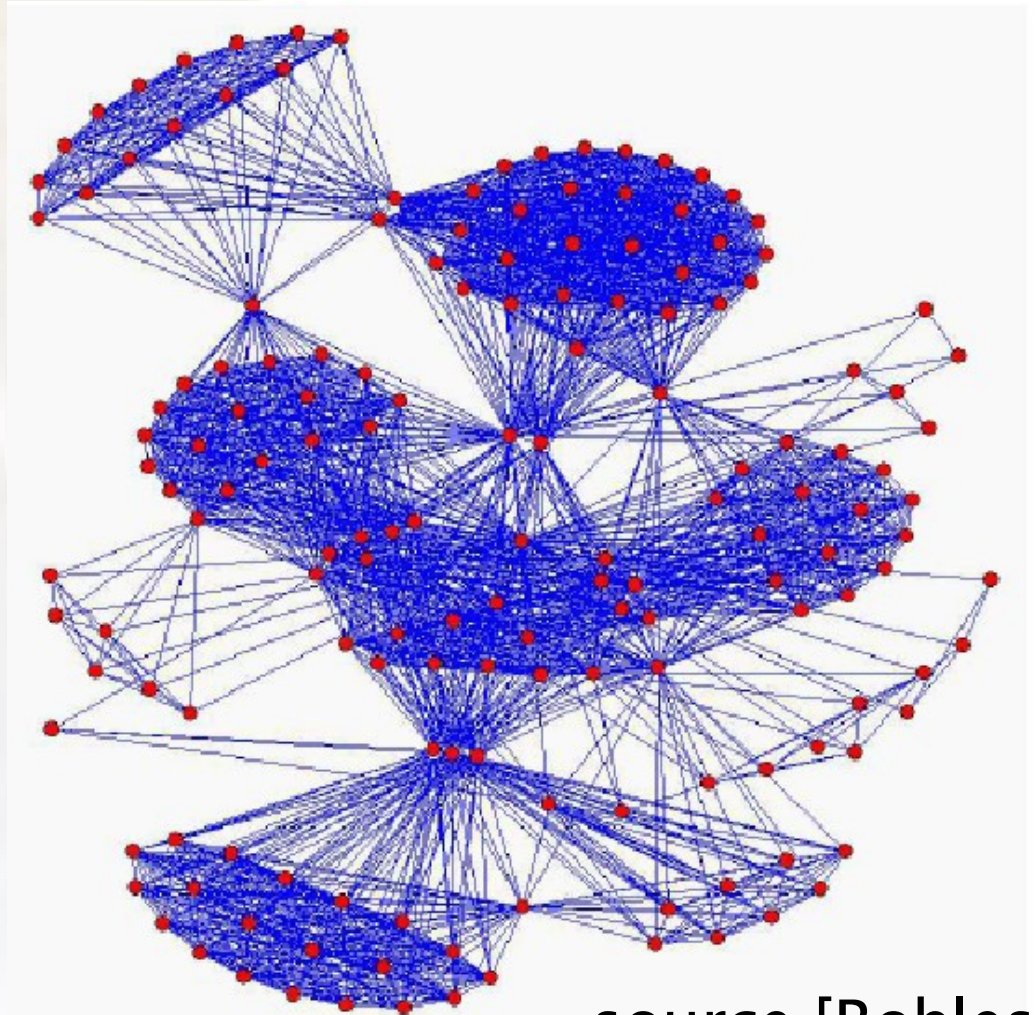
Socio-technical Analysis

- Structure of organisation = hints on software structure
- Analysis techniques for social networks

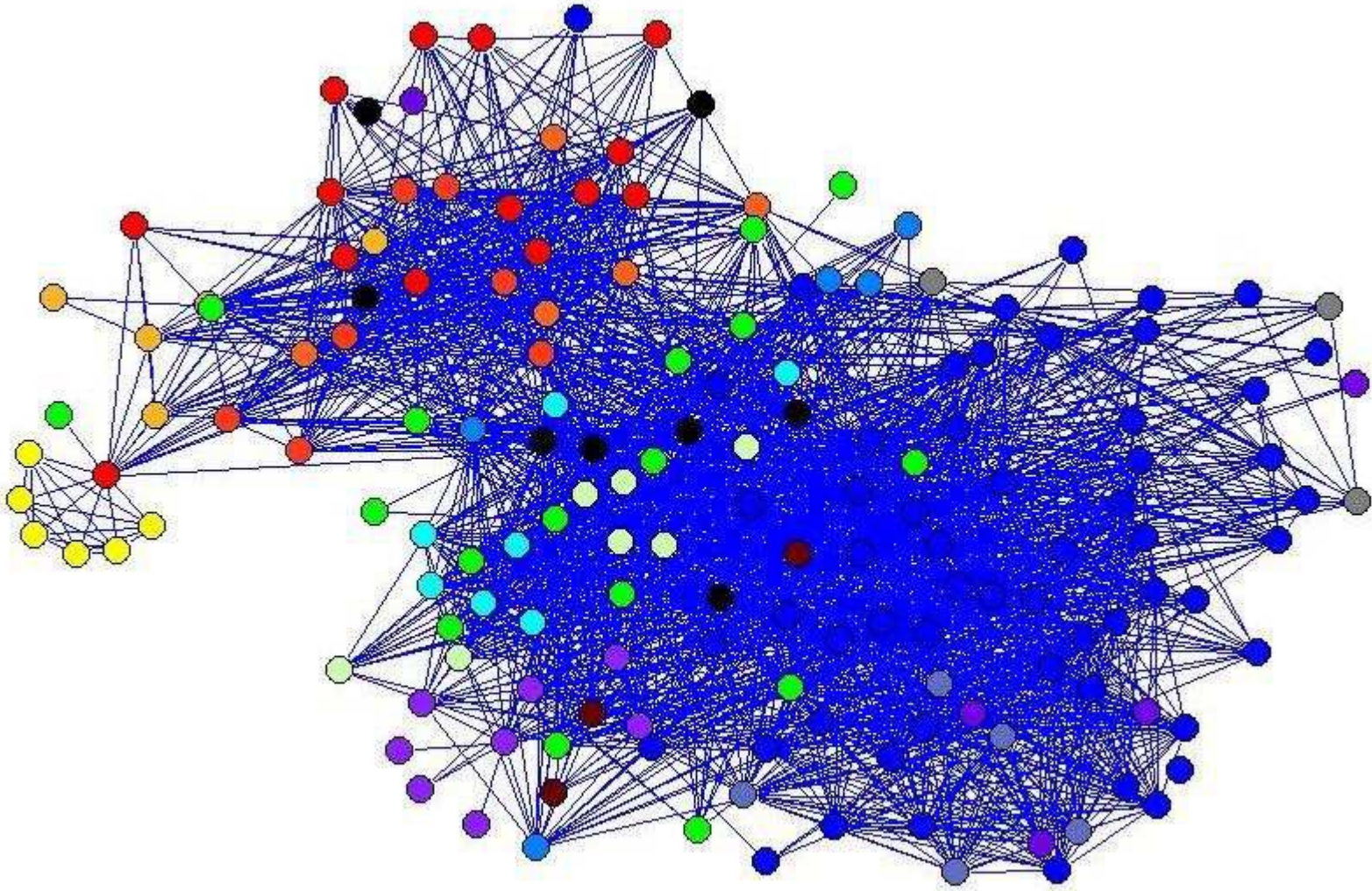
Developers network

**Linux 1.0
(1994)**

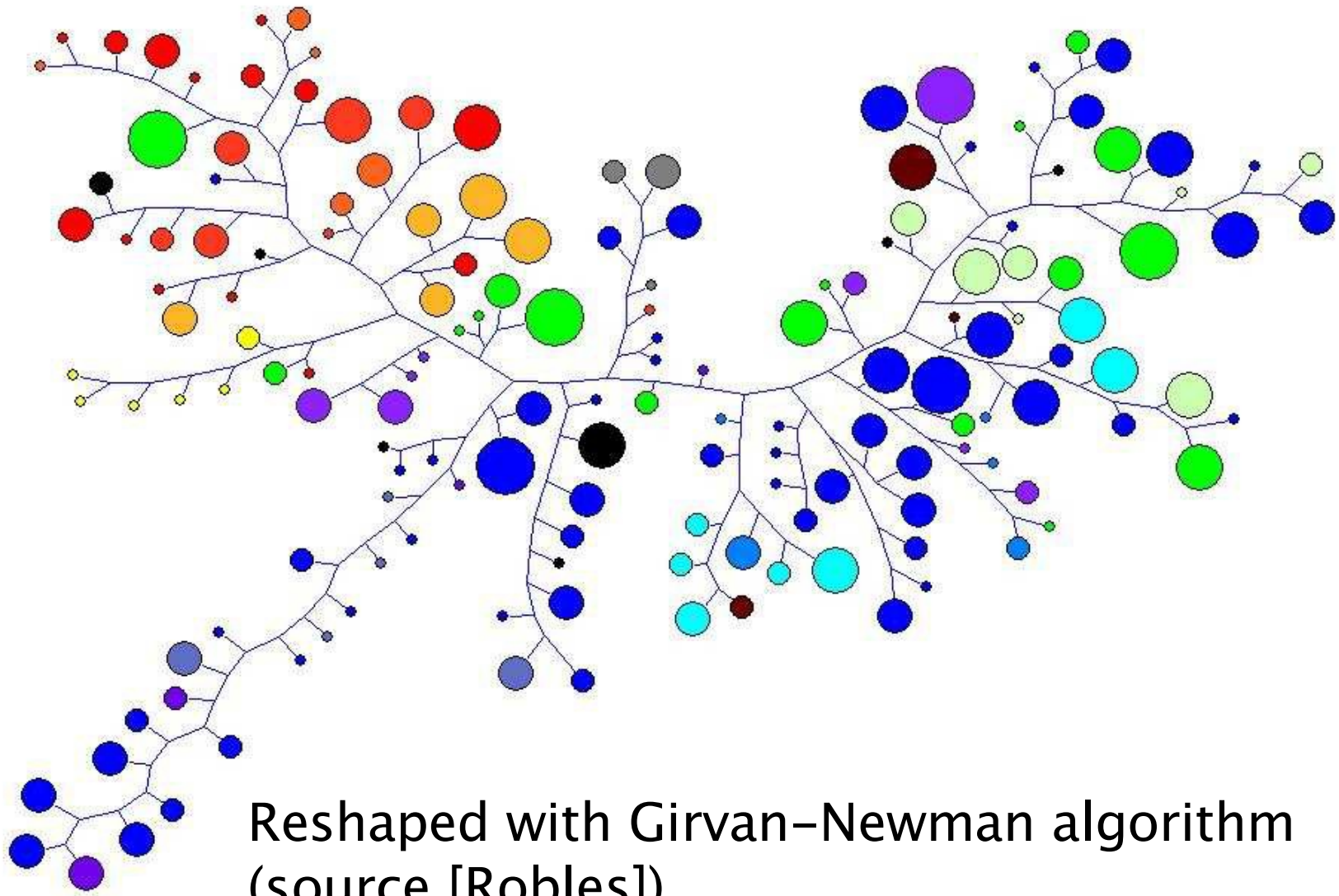
Developers
linked by
common
authorship to
same files



source [Robles]



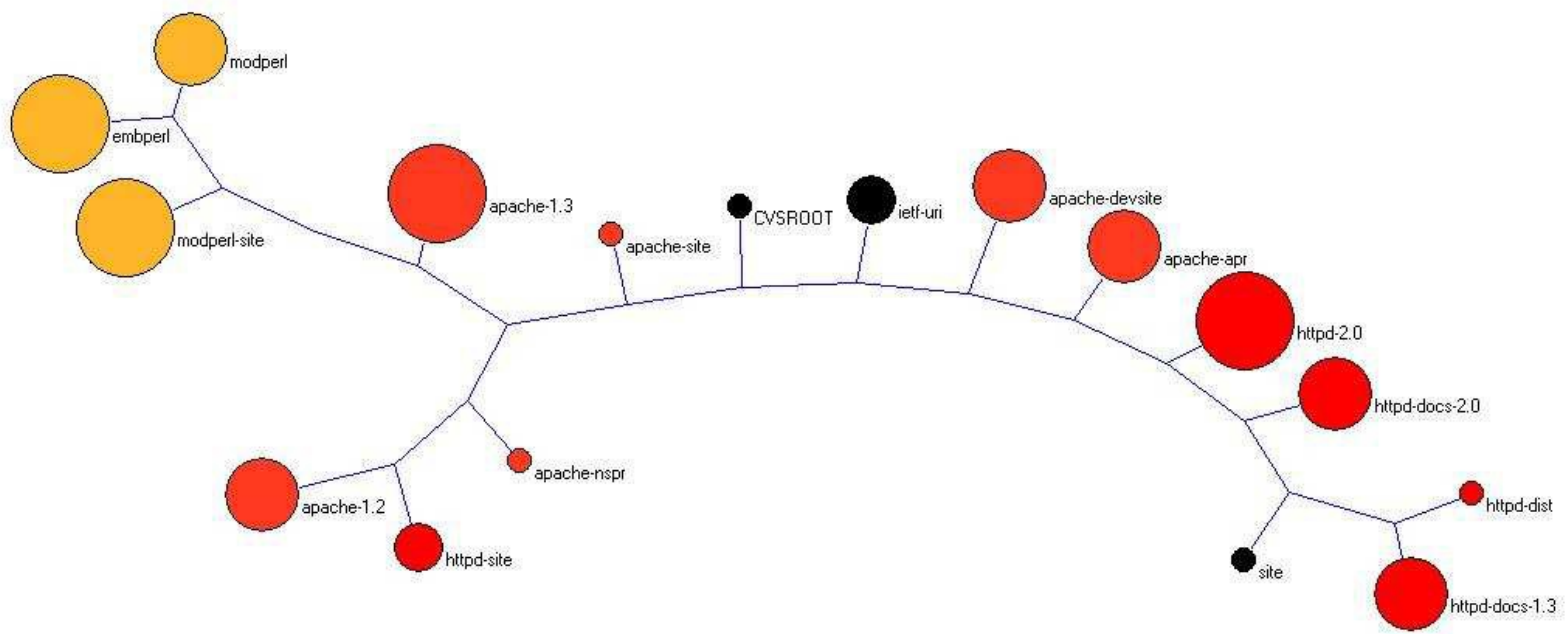
Classical analysis of Apache modules feb. 2004 (source [Robles])



Reshaped with Girvan–Newman algorithm
(source [Robles])

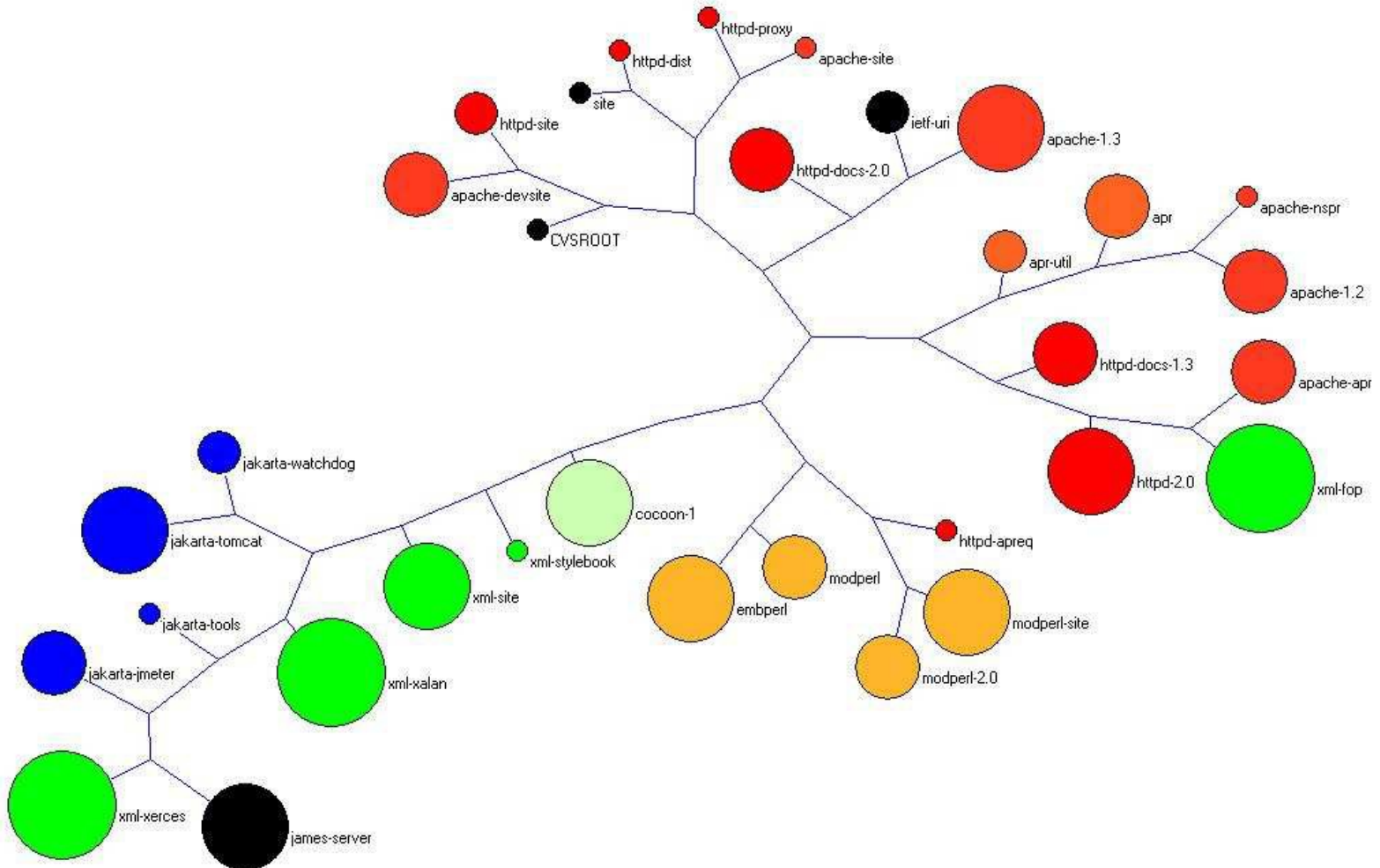


Apache 01/01/1999



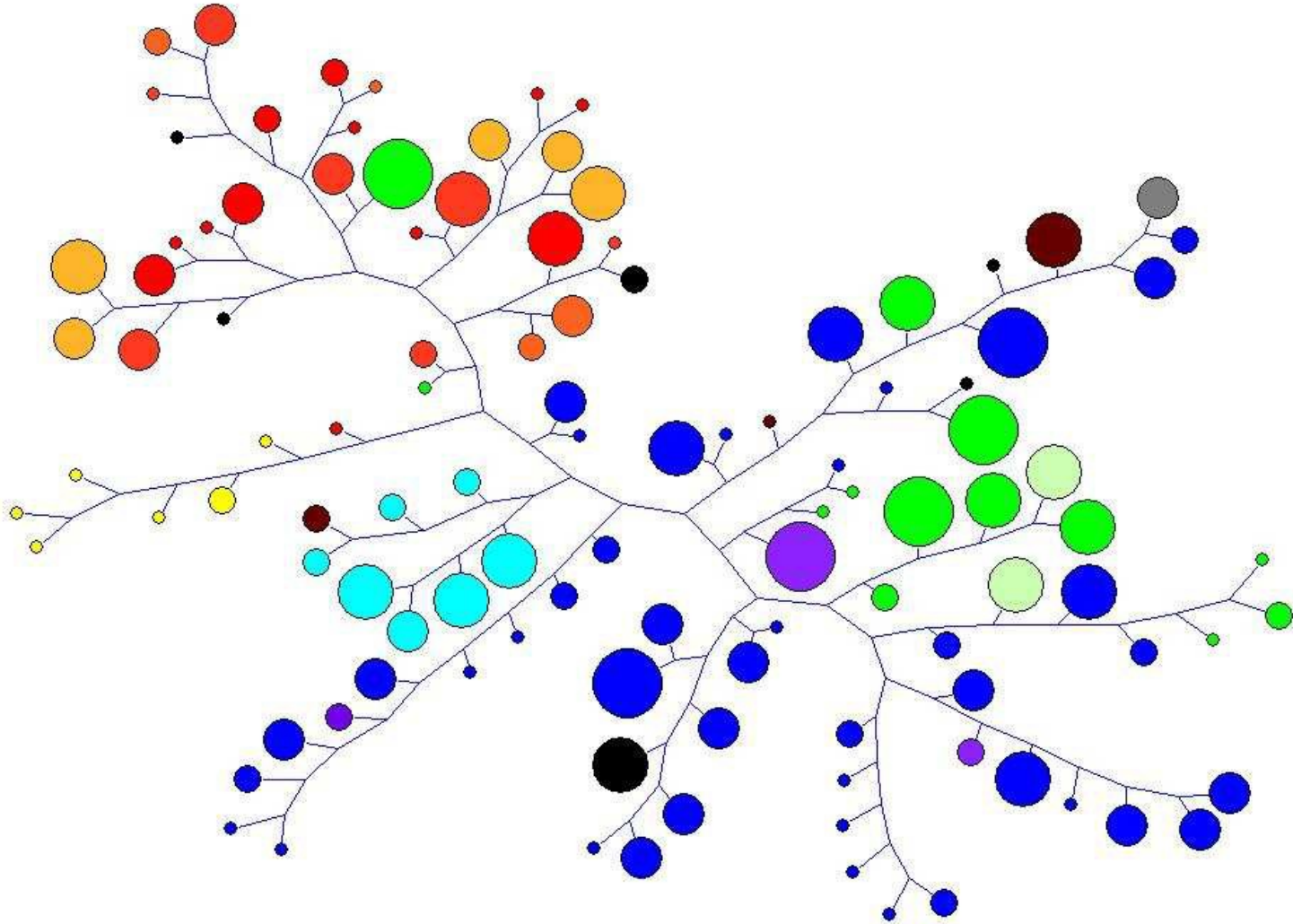


Apache 01/01/2000



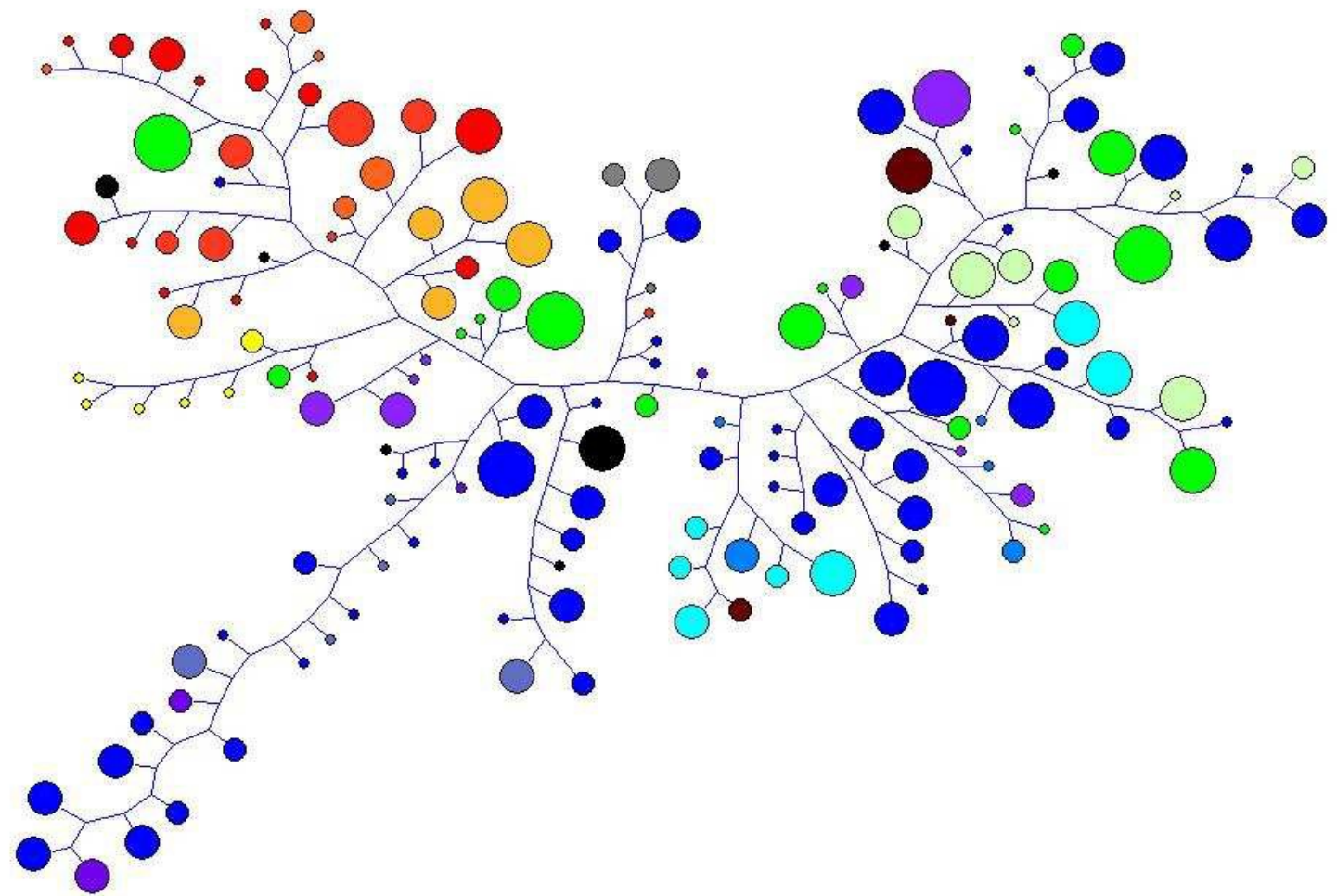


Apache 01/01/2002





Apache 01/02/2004



Valuing FLOSS

- Example: Debian 2.2 GNU/Linux (2001)
- Source lines of code: 55,201,526 (of which the Linux kernel forms under 6%)
- If written in a software company:
 - Estimated effort: 14,005 person years
 - Estimated schedule: 6.04 years (team of 2,318!)
 - Development cost: US\$ 1,891,990,000

(Source: "Counting potatoes" by Gonzalez-Barahona et al)

New SE era ?

- Public data sources are an important knowledge source for software projects
- Non-intrusive observation is possible for technical or social analysis
- Exhaustive analysis of huge amount of libre software projects is possible
- Possibility to define methodologies which can be applied in real-life projects


Limitations

- Some informations are not public (surveys)
- Some data sources are incomplete
- Necessary validation by the projects
- Respecting privacy



Dissemination

Calibration industry forum

- One of the ways to disseminate knowledge, and strategic decision criteria
- Targeted at big European Industry
- Not targeted at pure software firms
- Example of current members :
Philips Medical Systems, Eurocontrol, Telefónica, Thales, Vodafone, ...
- Link with academia and Commission 



OSS 2.0

The Transformation of Free/Libre/Open Source Software

From FLOSS to OSS 2.0

Process

FLOSS

OSS 2.0

Lifecycle

- | | |
|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <ul style="list-style-type: none"> ● Planning - "an itch worth scratching" ● Analysis part of general agreed-upon knowledge in s/w development ● Design modularity to achieve separation of concerns and reduce learning curve ● Implementation <ul style="list-style-type: none"> ○ Code ○ Review ○ Pre-commit test ○ Development release ○ Parallel debugging ○ Production release ● Organising principle having a tail-light to follow in bazaar ● Optimistic concurrency | <ul style="list-style-type: none"> ● Planning vendor-led purposive strategies ● Analysis & design more complex in vertical domains where business requirements not universally understood ● Implementation development process becoming less bazaar-like ● More holistic and planned development approach with paid developers considering marketing a consumable product |
|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

From FLOSS to OSS 2.0 cont'd

Product	FLOSS	OSS 2.0
Domains	<ul style="list-style-type: none"> ● Horizontal 'invisible' infrastructure 	<ul style="list-style-type: none"> ● More visible vertical domains
Business Models	<ul style="list-style-type: none"> ● Value-added/service-enabling ● Loss-leader/market-creating 	<ul style="list-style-type: none"> ● Hybrid models ● Value-added service enabling <ul style="list-style-type: none"> ○ Bootstrapping ● Market-creating <ul style="list-style-type: none"> - Loss leader - Dual product/licensing - Cost reduction - Accessorising ● Leveraging community development ● Leveraging open source brand
Support	<ul style="list-style-type: none"> ● Fairly haphazard bulletin boards, specialised firms - "the thrilling adventure of installing open source" 	<ul style="list-style-type: none"> ● Open Source Service Networks (OSSN) <ul style="list-style-type: none"> ○ Customer will pay for 'whole product' professional approach
Licensing	<ul style="list-style-type: none"> ● GPL, LGPL, Artistic, BSD, emergence of commercial MPL ● 'Viral' term used 	<ul style="list-style-type: none"> ● Plethora of licenses (from Microsoft alone!) ● 'Reciprocal' term used



Conclusion

OSS 2.0 Challenges – Research

- Transferring lessons to conventional development
 - Open sourcing an unknown workforce
 - Expanded role of users and altered user developer relationship
- Elaboration of business models



OSS 2.0 Challenges – Practice

- Balancing 'value creation' with 'acceptable community values'
- Stimulating development in vertical domains
- Implementing Open Source Service Networks and 'whole product' approach
- Safeguarding against IPR infringement
 - Indemnification of end users



Credits

- Many thanks to my Calibre colleagues
 - Dr Gregorio Robles–Martínez (Universidad Rey Juan Carlos, Spain),
 - Brian Fitzgerald (University Limerick), leader of the CALIBRE project,
 - Rishab Aiyer Ghosh (MERIT, Netherlands).



merci
thanks
xie xie