



# Free/Libre/Open Source software: engineering and economics

---

International Unnayan, BITS

Ranchi, Jharkhand, India, February 4, 2005

Rishab Aiyer Ghosh  
rishab@dxm.org

This presentation is based on work conducted in the projects CALIBRE and FLOSSPOLs, funded by the European Union's FP6 IST programme



# FLOSSPOLLS: EU Project

---

- Led by MERIT, University of Maastricht
- Largest EU-wide survey of government authorities on use of free software
- Major conference on November 18, The Hague: “Open Standards and Libre Software in Government”
- Large survey of developers and employers on skills development aspects of FLOSS



# CALIBRE: EU Project

---

- Led by University of Limerick, Ireland
- Project on open source software engineering, economics and industry
- MERIT, University of Maastricht is one of several partners, with a focus on measurement and economic aspects



# FLOSSWorld: EU Project

---

- Global follow-on project FLOSSWorld\*
- Led by MERIT, University of Maastricht
- Includes partners from Argentina, Brazil, Bulgaria, China, Croatia, India (CDAC/NCST), Malaysia, South Africa
- Includes survey of FLOSS use in govt, for skills development and in institutes of higher learning
- Includes country studies of software projects

\* FLOSSWorld is a project proposal selected for funding, but the contract is not yet signed



# What is “free software”?

---

Defined in 1984 as software with the:

- Freedom to run for any purpose.
- Freedom to change and modify.
- Freedom to copy and share.
- Freedom to share improvements.

All freedoms are commercial and non-commercial: you can sell “free software”



# “Open source” and “free software”

---

- English confuses free as in freedom with free as in no price
- “Open source” was coined in 1997 as a business-friendly term for “free software”
- Both terms have official definitions, and refer to the same software
- A political movement around free software, though, emphasises freedom



# “Libre” and FLOSS

---

- French, Spanish etc avoid the English confusion: libre software
- Free/Libre/Open Source Software – created as acronym for FLOSS project\*
- Since the success of the project, FLOSS has become a widely used “compromise” term in Europe, Latin America and Asia

\*FLOSS was funded by the EU FP5, and led by Infonomics/MERIT at the University of Maastricht



# Selling free software

---

- IBM, HP, SAP, Oracle, Novell are companies with large FLOSS-related revenues in billions of \$ annually
- Red Hat, MySQL, Mandrake are companies that work mainly with FLOSS
- They charge for software CDs, support, administration, customisation, hardware; using FLOSS as a platform



# Examples of FLOSS

---

- GNU/Linux: operating system based on the Linux kernel + other free software including the GNU system (from gcc and libraries to bash and emacs...), Apache, Mozilla, Sendmail, etc
- Distributions include Red Hat, Debian, Novell/SuSE... various Indian-language distributions

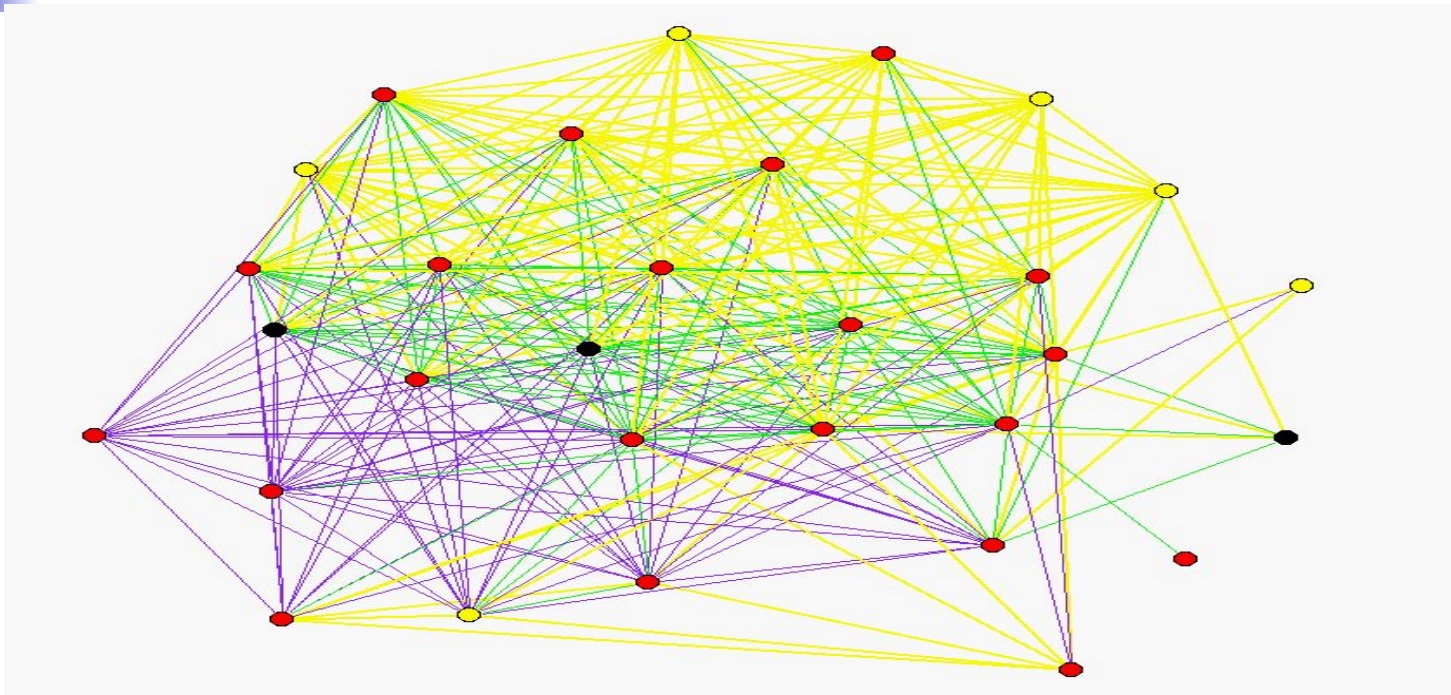


# How is FLOSS developed?

---

- The Linux kernel was first written by Linus Torvalds, a student in Finland in 1990, and released over the Internet under the General Public Licence (GPL)
- GPL says: this provides the 4 freedoms, but if you modify it, your modified version must be free software too
- GPL + Internet allowed many authors...

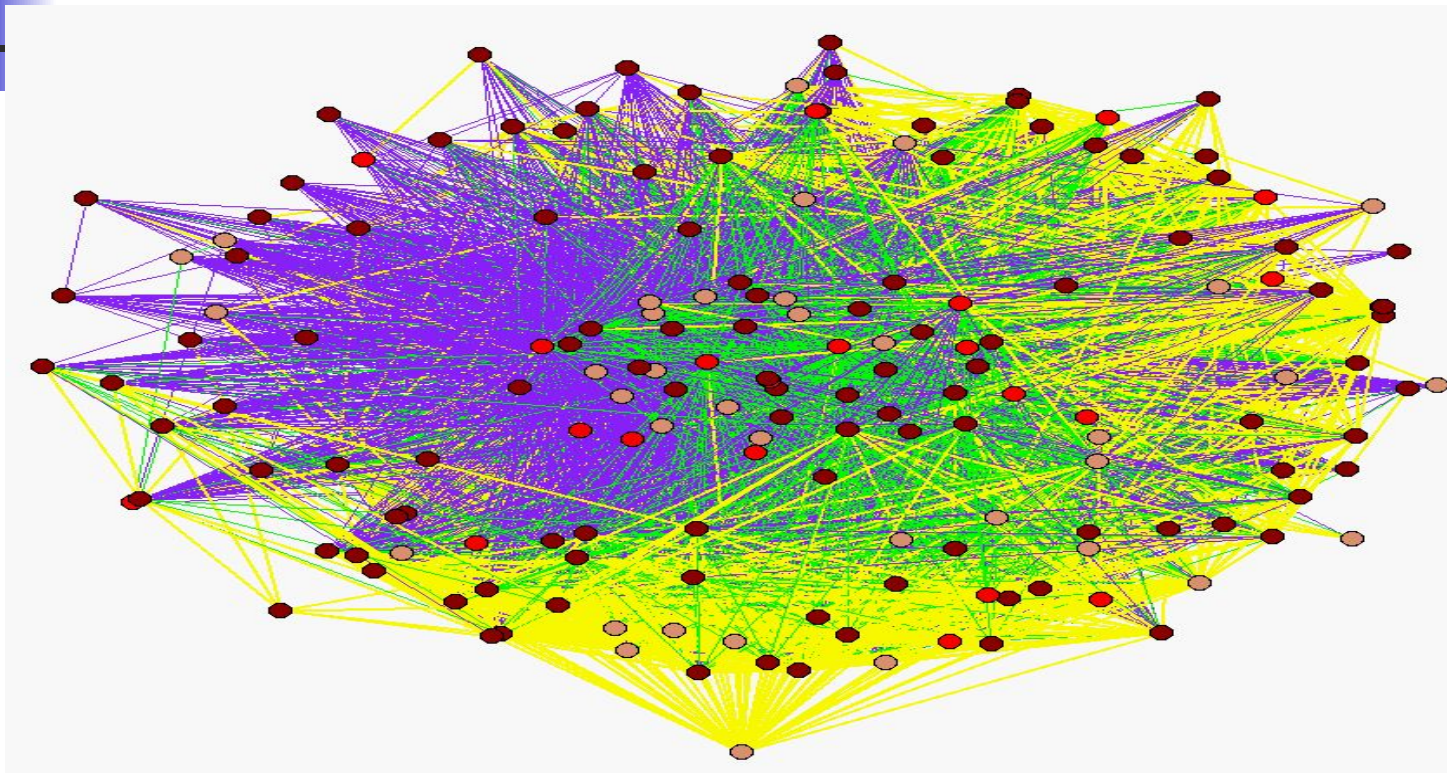
# Linux kernel v1.0: 158 authors



1994. Nodes are 30 modules. Arcs represent **common authors**, code dependencies, or **both**

(Source: "Nature and composition... ", Ghosh & David)

# Linux kernel v2.5.25: 2263 authors



2002. Nodes are 169 modules. Arcs represent **common authors**, code dependencies, or **both**

(Source: "Nature and composition... ", Ghosh & David)



# 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 Debian was 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)



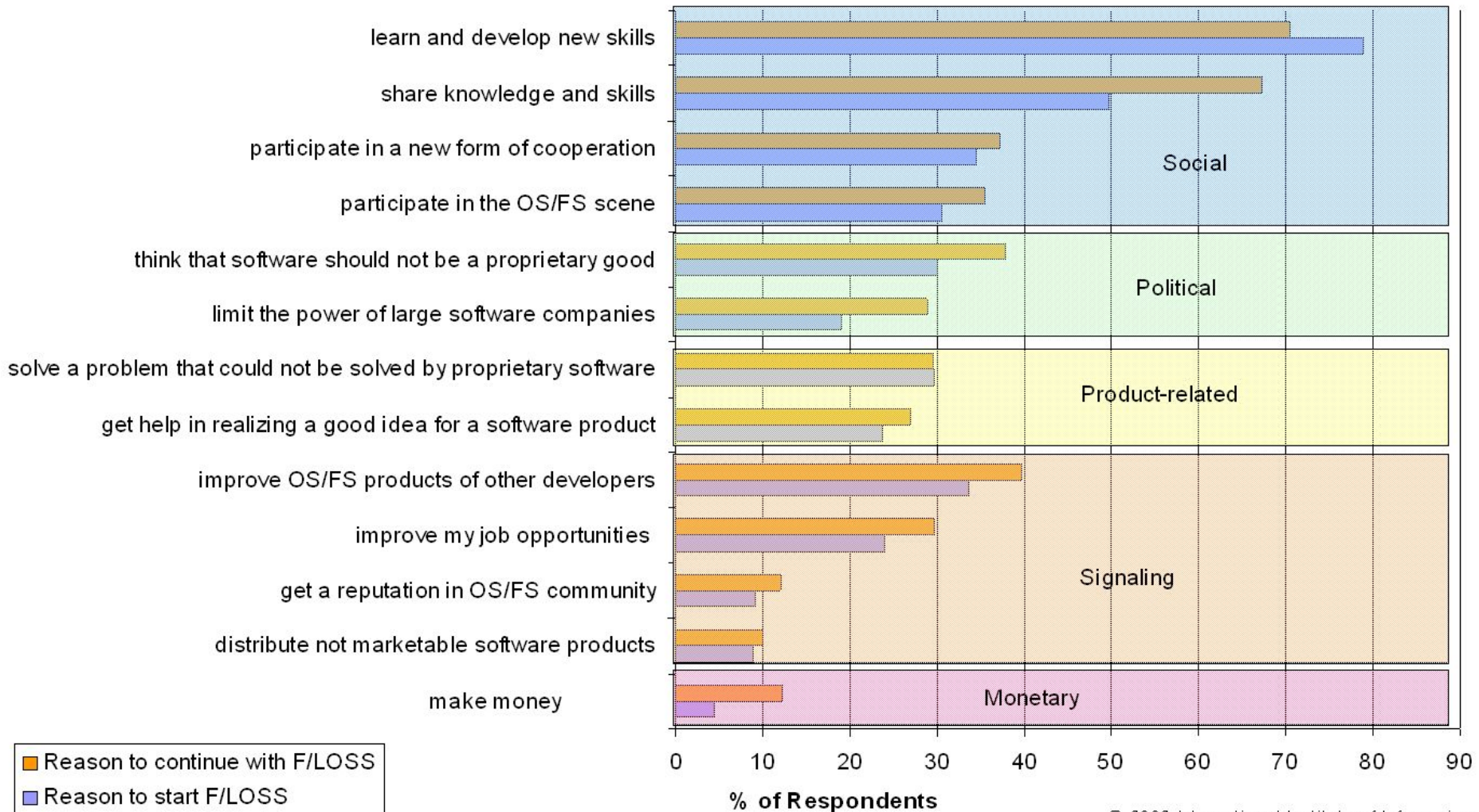
# Who develops FLOSS?

---

- Young (av. age 26) and largely male community
  - High educational level
  - 44% IT sector, 15% IT students, 5% other students, 4% other engineers
  - Most (59%) are married or have partners
  - Most employed (65%) or self-employed (14%)
  - PhD 9%, University degree 61%, high-school 25%
  - 49% do not contribute more than 5 hours/week
- Start young: 14! or old: 73! Average starting age 22

(Source: "FLOSS Final report", Ghosh et al)

# Why develop FLOSS?





# FLOSS develops local skills

---

- FLOSS encourages not only passive “use” but active participation in the creative process
- FLOSS provides a very low barrier to entry for creativity – you don’t have to be creative but if you want to, you easily can



# Learning skills – then sharing!

---

- 78% of developers join the FLOSS community “to learn and develop new skills” (70% continue for this reason)
- 67% of developers continue their participation in the FLOSS community “to share ... knowledge and skills”

Source: Free/Libre/Open Source Software (FLOSS) Study of Developers



# These skills have economic value

---

- 30% of developers participate in the FLOSS community “to improve ... job opportunities”
- Over 30% of developers derive income directly through their FLOSS work
- A further 20% derive indirect income as a result of their FLOSS work
- 18% got job because of FLOSS experience

Source: Free/Libre/Open Source Software (FLOSS) Study of Developers



## Employers appreciate this...

---

- 36% of organisations “totally” or “somewhat” agree that employees can work on FLOSS projects on employer time
- These are not software companies! 16% of low IT-intensity companies (retail, automobiles, tourism, construction) “totally agree” with this

Source: Free/Libre/Open Source Software (FLOSS) Study of Users



...but don't pay for it.

---

- FLOSS communities are like informal apprenticeships – but apprentice/students and master/teachers contribute their own time for free
- Any cost (time, effort) is borne voluntarily by the participants themselves and not paid for directly by those who benefit (employers, society at large)



...but don't pay for it.

---

- Sectoral benefits: Universities may have formal computer training during computer science degree courses, but perhaps not in other subjects (biology...)
- FLOSS usage provides students of other subjects to informally learn computer skills, programming skills and enhance their competence in their formal training



# But do we all want to program?

---

- How will we know, unless we can try?
- HTML is a programming language – the web only took off because it was open, so people could learn to write their own sites just by copying and changing other sites
- “Programming” covers a very broad range of skills from HTML to C; FLOSS allows entry at any degree with little investment in time or effort



# But do we all want to program?

---

- In a proprietary environment, you have to decide to be a programmer, then buy development software, then spend lots of time and effort – all of which is a risk and entry barrier
- With FLOSS, you can tinker. You don't need to buy tools. You can use them to the extent you choose.



# But do we all want to program?

---

- Learning skills in FLOSS, you risk losing only your time and effort
- However, since the barrier to entry is low (HTML!) you can control the degree of your investment – paddle at the shallow end or dive in deeper.
- In proprietary environments, the dividing line between user and developer is much sharper – the pool has only a deep end, you have to dive in or stay out altogether.



# (very) Preliminary survey findings

---

FLOSSPOLLS survey on skills learnt and impact on employability: separate questions sent to developers (worldwide) and to employers (EU)

Developer survey:

- 350 respondents
- Limited to respondents of the previous FLOSS survey of developers (2002)



# (very) Preliminary survey findings

---

- Technical skills
  - New developers learn various skills
  - Experienced developers learn to write reusable code, to detect and fix bugs
- Management skills
  - Most respondents learn to “plan work and stick to a schedule”!
  - And to coordinate their work with others



# (very) Preliminary survey findings

---

- Legal skills
  - Most developers learn “a lot” about copyright, patents, and licensing
  - Employers agree that such skills are better learned in FLOSS communities compared with an ICT degree course
- General skills
  - Non-English speakers improve their English



# (very) Preliminary survey findings

---

In comparison with formal ICT courses:

- FLOSS provides a better, practical learning environment for many technical skills:
  - Writing re-usable code & debugging
  - Working with code written by others
- FLOSS provides a better learning environment for most legal and teamwork skills, which are rarely taught in formal ICT courses



# Skills and economic growth

---

- **Skills development: “the ability to create”**  
FLOSS is a training environment that increases the earning capacity of community participants without any explicit investment in training: a novel form of technology transfer
- **Economic growth: “ability to add value”**  
FLOSS allows local entrepreneurs to provide a greater share of total value added, thus retaining a greater share of profits within the local economy



# Local value addition: proprietary

---

- **Building over a platform**

This applies equally to any platform, which is simply used as a (non-modifiable) base on which new services or software are built: 100% of the added value is local

- **Sales commissions**

Something which is rarely possible with free software, but also represents little value. Only the commission is retained locally, which is a small part of the total value.

- **Support, integration, customisation...**

Local value addition limited, as “deep” (high-value) services require “deep” access – only the proprietor has it.



# Local value addition: FLOSS

---

- **Building over a platform**

As with proprietary software, free software platforms can be used as a (modifiable!) base on which new services or software are built: 100% of the added value is local

- **Sales commissions**

Rarely possible with free software, but also represents little value. However, the entire "sale price" can be retained locally, as no proprietor has to be paid a royalty or licence.

- **Support, integration, customisation...**

Local value addition extensive, as "deep" access is available. 100% of such services can be provided locally, retaining 100% of the value locally.



# The importance of customisation...

---

- Custom or in-house software represents about 67% of total software produced (in the US; probably more elsewhere)
- If based on free software, custom solutions greatly benefit the solutions provider who captures 100% of the total value, not just the value added locally – no royalties/licences paid



# Code re-use, higher service levels

---

- Free software allows providers to reuse code rather than build from scratch, and to reuse a huge base of code written by others
- Re-using (and modifying) allows the creation of much better end-user solutions for the same effort than writing from scratch
- Put together, this provides better value for money for customers and better profit margins for local service providers



# “Deep” support, more local value

---

- Local companies are limited in the integration and support services they can provide for proprietary software
- Deep support: fixing software bugs, customising it to user requirements, or integrating extensively with other software requires deep access.



# “Deep” support, more local value

---

- Deep access to proprietary software is controlled by the proprietor (limits access or requires royalties, diminishing value retained locally)
- Deep access to free software is available to anyone – limited only by their skills. This allows every provider to potentially provide deep support services, and retain 100% of the value.



# Skills and economic growth

---

- **Skills development: “the ability to create”**  
FLOSS is a training environment that increases the earning capacity of community participants without any explicit investment in training: a novel form of technology transfer
- **Economic growth: “ability to add value”**  
FLOSS allows local entrepreneurs to provide a greater share of total value added, thus retaining a greater share of profits within the local economy



## • Building local ICT competencies

---

- Be passive users of “black-box” software or active participants in global ICT?
- Being active requires being able to create, locally – and choose with the least barriers the level of creativity
- Skills development requires access to the ability to create – you don’t have to be a programmer, but you should have the choice.
- Relative local value addition is much higher with free software, as compared to proprietary (where the vendor controls and provides the most value)



# What can you do?

---

- Start/contact Linux User Groups (LUGs)
- Contact FLOSS projects in India and abroad
- Read and (eventually) participate in discussion groups for specific projects that interest you
- Submit bug reports
- Download, read, experiment with source code
- Submit bug fixes (your own code)
- Contribute software to the community



# FLOSS participation from India

---

- Indians have key positions in some global free software projects, including Perl
- Numerous projects originating in India – mainly localisation efforts (Indinux, bangalinux etc)
- India has active user and developer groups, and the Free Software Foundation India (only FSF outside the US and Europe)
- FLOSSWorld will study Indian FLOSS further



## In conclusion

---

- FLOSS use is rapidly growing worldwide
- This is driven by demand from government and from industry: FLOSS is viable business
- FLOSS appears to have positive implications for skills and employment generation
- FLOSS participation is open to anyone
- FLOSS participation is the fastest way for Indians to develop international links



## More information

---

FLOSSPOLS: <http://flosspols.org>

CALIBRE: <http://www.calibre.ie>

### References:

- FLOSS report: <http://flossproject.org/report/>
- “Nature and composition...” (Ghosh & David):  
<http://dxm.org/papers/licks1/>
- “Counting Potatoes” (Gonzalez-Barahona et al):  
<http://libresoft.gsync.urjc.es>