Detecting Agility of Open Source Projects Through Developer Engagement

Paul Adams, Adriaan de Groot

Research and Development

Sirius Corporation plc, UK

Andrea Capiluppi

Centre of Research on Open Source Software

Department of Computing & Informatics

University of Lincoln, UK

A Quote...

“ Dude, we are so CMM 0! ”

Adriaan de Groot

Vice President, KDE e.V.

Centre of Research on Open Source Software

UNIVERSITY OF LINCOLN

About This Talk

- **Context** and **Objective**
- **Mean Developer Engagement**
  - Grace Period
  - Data Gathering and Processing
- **Evaluation**
- **Results and Analysis**

Centre of Research on Open Source Software

UNIVERSITY OF LINCOLN

Context

- Open Source and Agility are fundamentally **different**
  - Open Source: *licensing* model, philosophy
  - Agility: *process* model, philosophy

- ...but OSS and Agile are also often **similar**
  - feature driven
  - lightweight
  - *sprinting*

Centre of Research on Open Source Software

UNIVERSITY OF LINCOLN

Objective

- Objective: empirically evaluating the *agility* of OSS processes
  - No **previous** empirical studies of agility within OSS
  - No **comparative** studies:
    - Are some OSS projects more agile than others?
    - Do some OSS repositories lead to more agility than others?

Centre of Research on Open Source Software

UNIVERSITY OF LINCOLN

Agility

- Agility (from Dat06)
  - Criticality of the product
  - **Dynamism of the team**
  - **Effort required along the duration**

↓

Mean Developer Engagement

Centre of Research on Open Source Software

UNIVERSITY OF LINCOLN

Mean Developer Engagement

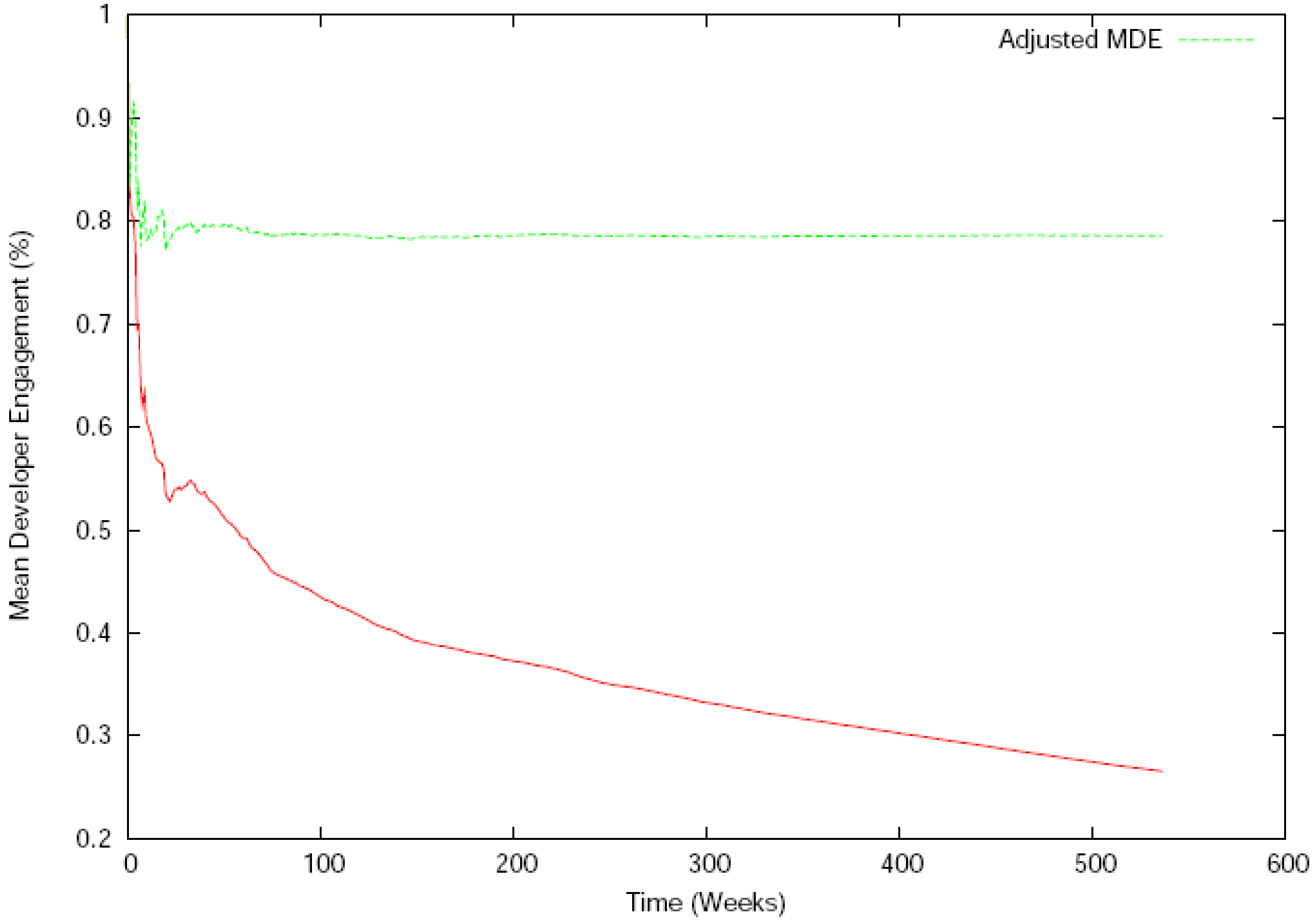
$$mde = \frac{\sum_{i=1}^{n} \left(\frac{\mathrm{dev}(active)}{\mathrm{dev}(total)} \right)_i}{n}$$

- A measure of how effectively an Open Source project makes use of its **available human resources**
- For each week:
  - Find the ratio of active to total developers and average over the lifetime of the project

Centre of Research on Open Source Software

UNIVERSITY OF LINCOLN

Example – KDE Project



Centre of Research on Open Source Software

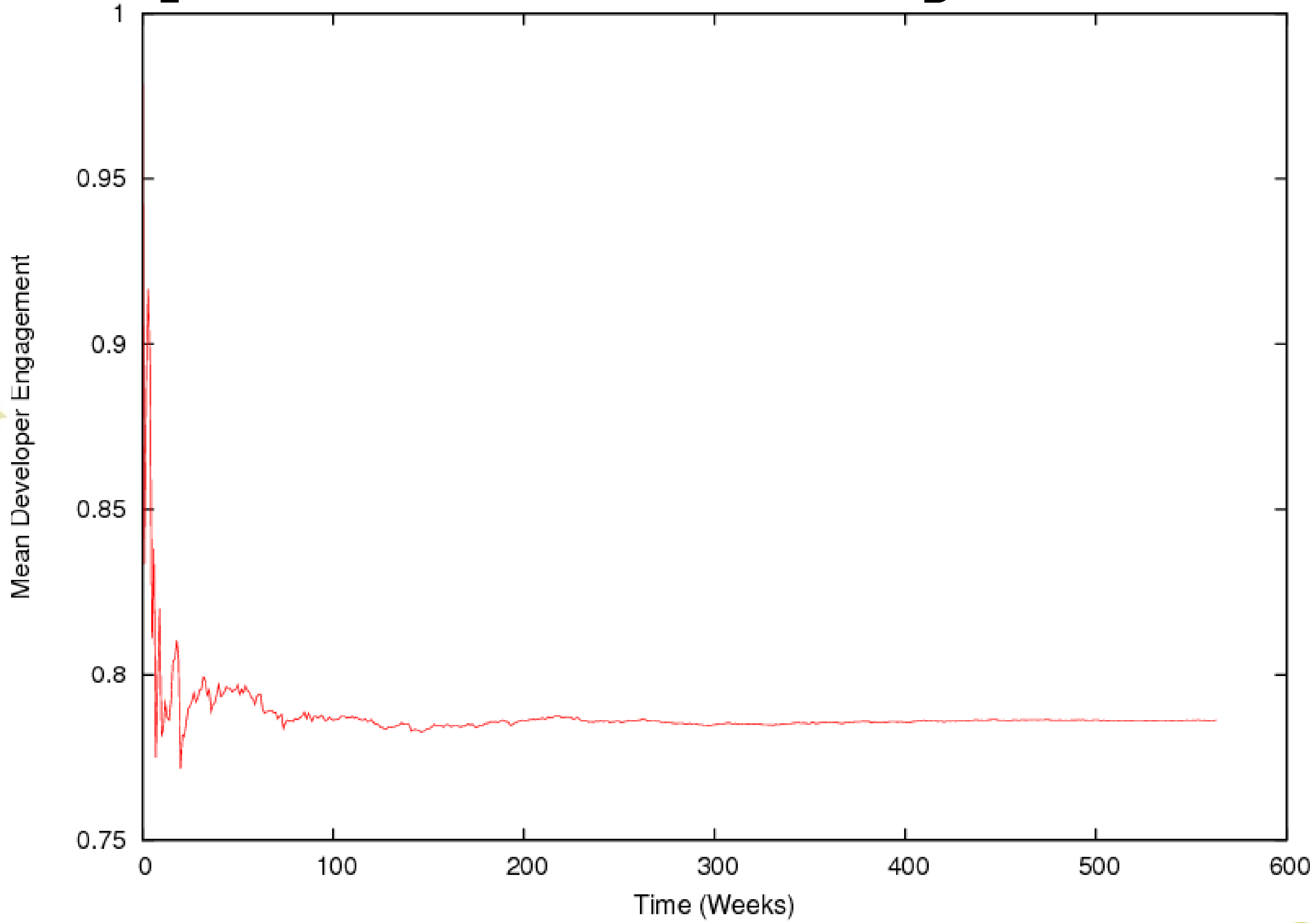
UNIVERSITY OF LINCOLN

Grace Period

- Using the total number of Version Control System (VCS) accounts for *dev(total)* is suboptimal
  - What about *“dead”* accounts?
- **Grace period** allows for developers leaving the project and for their potential return too

Length Of Service	Grace Period
1024	20
520	15
416	12
260	10
208	8
104	6
52	4
24	2
≤ 23	1

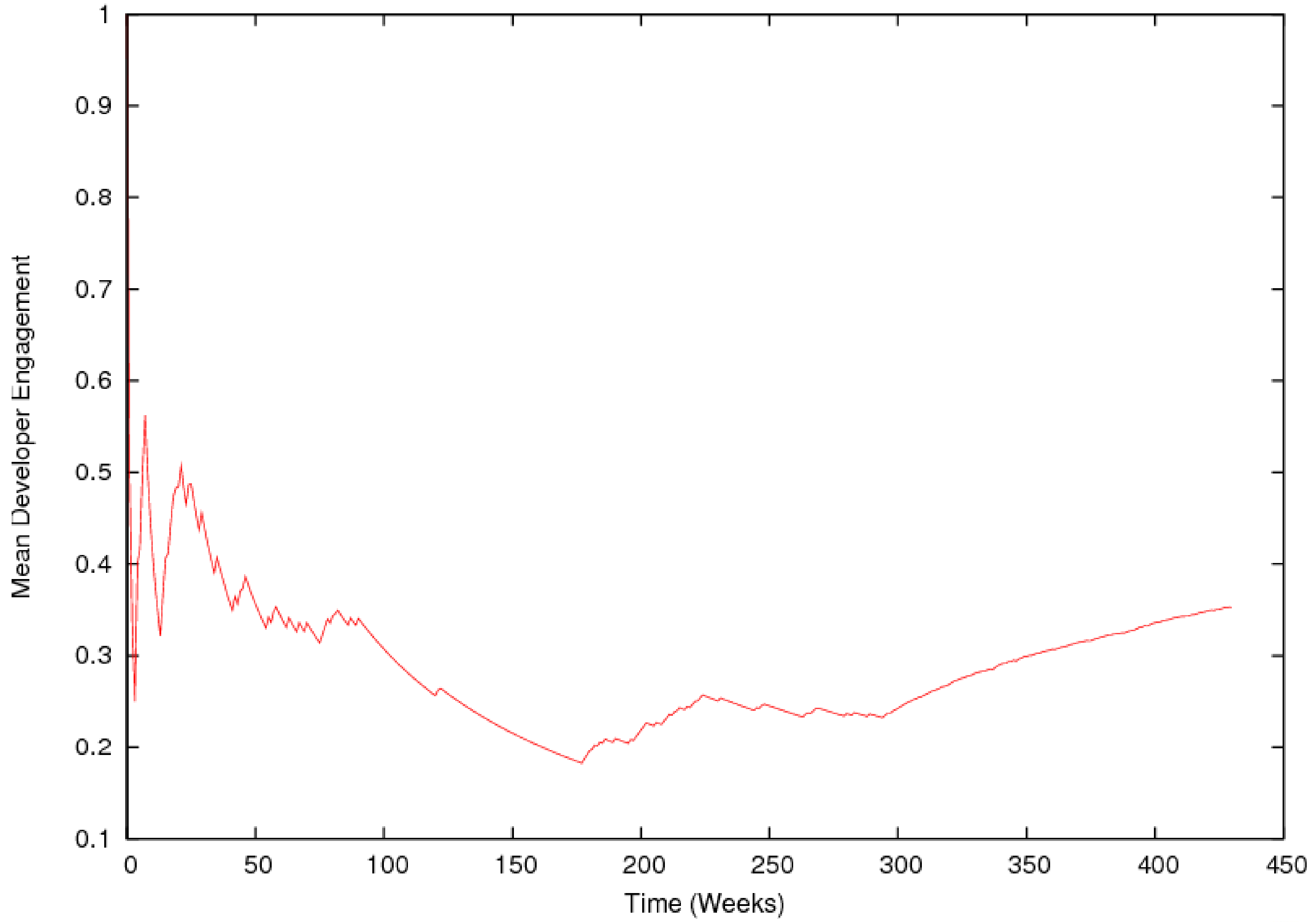
Example – KDE Project 2



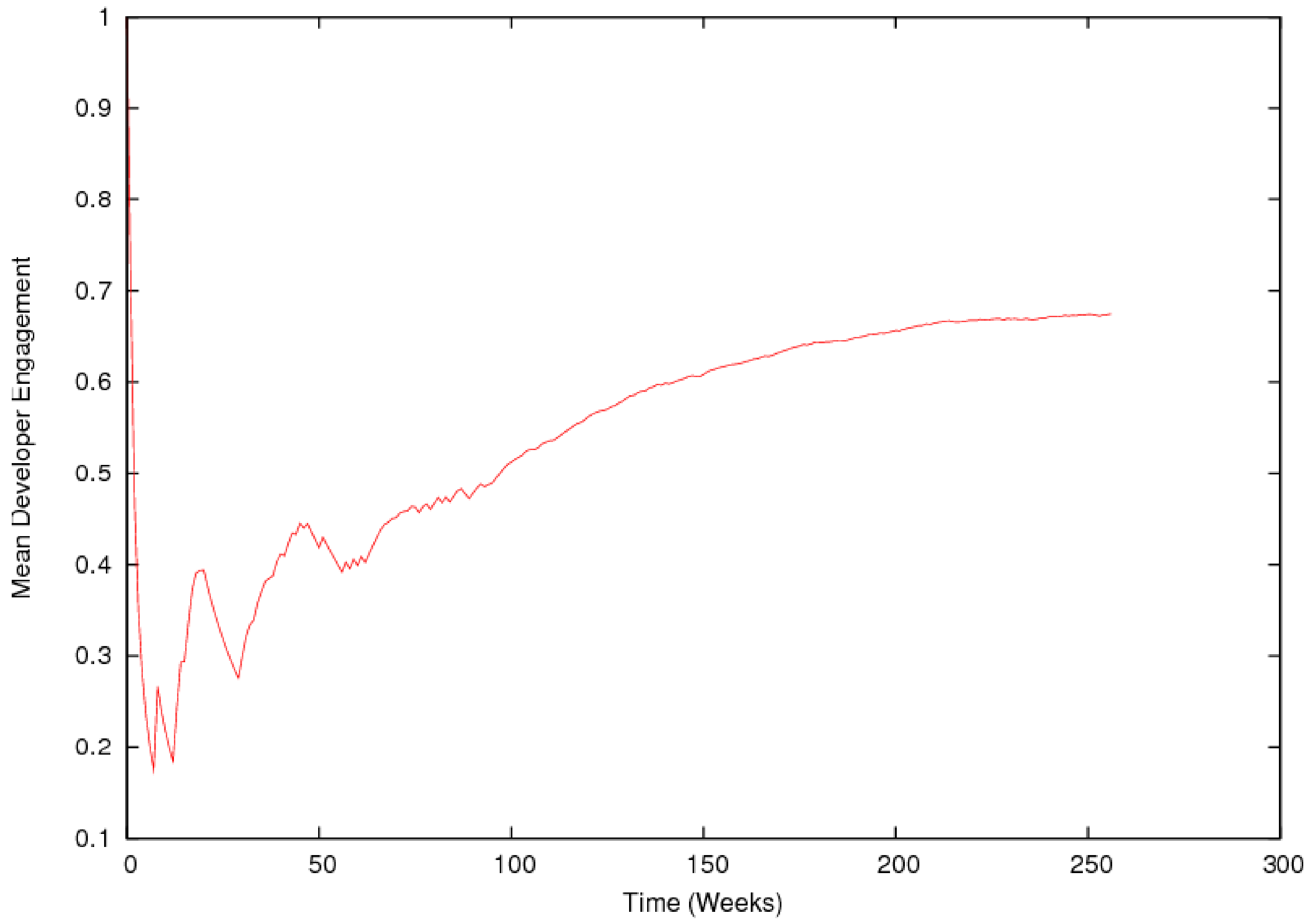
Centre of Research on Open Source Software

UNIVERSITY OF LINCOLN

Example - Evince



Example - PyPy



Centre of Research on Open Source Software

UNIVERSITY OF LINCOLN

Data Gathering and Processing

- For MDE we need to know
  - Who in the project did *something*?
  - When did they do it?
- VCS logs provide this information in a structured manner (e.g. XML)
- This structured documentation can be processed for data extraction (e.g. Python + SAX)

Centre of Research on Open Source Software

UNIVERSITY OF LINCOLN

Empirical Evaluation

- Goal, Question, Metric:
  - *Goal*: To assess the degree of agility displayed by Open Source projects.
  - *Question:* Are projects within KDE statistically different from those in SourceForge with regards to MDE?
  - *Metric:* MDE is applied across each project's lifetime.

Centre of Research on Open Source Software

UNIVERSITY OF LINCOLN

Results

KDE					*SF.net*				
Project	Start	i	MDE	Effort	Project	Start	i	MDE	Effort
dolphin	21-11-06	54	0.6799	36.7146	askcms	29-06-06	1	1.0	1.0
k3b	26-03-01	349	0.5961	208.0389	awdotnet	24-05-07	1	1.0	1.0
katomic	29-06-99	438	0.3340	146.2920	dvdshop	31-03-06	1	1.0	1.0
kcalc	13-04-97	554	0.4307	238.6078	hivex	16-07-07	1	1.0	1.0
kgeography	07-03-04	38	0.5386	20.4668	interaction	04-03-07	1	1.0	1.0
kig	15-04-02	294	0.6878	202.2132	kuragari	13-01-07	1	1.0	1.0
kivio	02-12-00	365	0.5320	194.1800	kyrios	03-07-06	74	0.5634	41.6916
kmail	18-01-03	254	0.6730	170.9420	map	06-11-05	50	0.3780	18.9000
kmoon	27-09-98	478	0.2913	139.2414	neuralbattle	14-06-06	74	0.6803	50.3422
knotes	30-06-97	544	0.4638	252.3072	opulus	25-07-07	10	0.4931	4.9310
kolourpaint	10-10-03	214	0.6269	134.1566	pwytter	09-07-07	1	1.0	1.0
konqueror	09-02-99	459	0.6610	303.3990	pyaws	11-04-06	56	0.2514	14.0784
konsole	28-10-98	474	0.6109	281.5666	rejuce	02-08-06	10	0.5554	5.5540
kontact	18-01-03	254	0.5867	149.0218	rlcyber	02-07-06	17	0.7612	12.9404
kopete	02-01-02	308	0.7142	219.9736	shareaza	02-06-04	182	0.7830	142.5060
kscd	04-07-97	542	0.4962	268.9404	stellarium	12-07-02	280	0.6183	173.1240
kspread	18-04-98	502	0.6216	312.0432	tclshp	04-10-06	1	1.0	1.0
ksudoku	03-03-07	39	0.6530	25.4670	tsg	23-03-07	19	0.6036	11.4684
kteatime	16-04-99	450	0.3299	148.4550	wxpropgrid	16-04-07	31	0.9968	30.9008
marble	29-09-06	61	0.6321	38.5581	xml-copy-editor	16-08-07	14	0.7731	10.8234

Analysis

- 20 projects randomly selected from each of KDE and SF.net
- Wilcoxon Test applied to MDE data
  - **Null hypothesis** (H_0): The samples have similar levels of MDE
  - **Alternative hypothesis** (H_1): KDE shows greater level of MDE
- Both rejected: Note correlation between MDE of 1.0 and *i=0*
- “Adjustments” made according to *i*

Conclusions, Further Work

- MDE measures a project's utility of its human resources over time
- As such, MDE is an *implicator* of agility, not a measure
  - As indicated by the results for KDE vs. SF.net
- MDE still needs to be compared with “known” agile data before formal introduction.

Centre of Research on Open Source Software

UNIVERSITY OF LINCOLN

Ackowledgements

CALIBRE
Coordination Action for LIBRE software

SQO OSS

Sirius

CROSS
Centre of Research on Open Source Software

UNIVERSITY OF LINCOLN