

EUROPEAN ORGANISATION
FOR THE SAFETY OF AIR NAVIGATION



EUROCONTROL EXPERIMENTAL CENTRE

POTENTIAL OF OPEN SOURCE SOFTWARE FOR AIR TRAFFIC MANAGEMENT

EEC Report No. 406

Project OSIFE

Issued: November 2006

REPORT DOCUMENTATION PAGE

Reference: EEC Report No. 406		Security Classification: Unclassified	
Originator: OSIFE project Innovative Research Department (INO) EUROCONTROL-EEC		Originator (Corporate Author) Name/Location: EUROCONTROL Experimental Centre Centre de Bois des Bordes B.P.15 F - 91222 Brétigny-sur-Orge Cedex FRANCE Telephone : +33 (0)1 69 88 75 00 Internet : www.eurocontrol.int	
Sponsor: OSIFE project (EUROCONTROL-EEC-INO) and CALIBRE project (EU FP6, www.calibre.ie)		Sponsor (Contract Authority) Name/Location: EUROCONTROL Agency 96, Rue de la Fusée B-1130 Brussels Telephone: +32 2 729 9011 www.eurocontrol.int	
TITLE: POTENTIAL OF OPEN SOURCE SOFTWARE FOR AIR TRAFFIC MANAGEMENT			
<u>Editors</u>		<u>Contributors</u>	
<p>Marc Bourgois, EUROCONTROL-EEC, FR Andrea Deverell, University of Limerick, IE Pr. Brian Fitzgerald, University of Limerick, IE Jean-Luc Hardy, EUROCONTROL-EEC, FR John O'Flaherty, Mac, IE John Seifarth, Waw, BE</p>		<p>Carlos Garcia Avello, EUROCONTROL, BE Garfield Dean, EUROCONTROL-EEC, FR Arnoud Engelfriet, PHILIPS, NL Joseph Feller, University College Cork, IE Jean-Dominique Frayssinoux, ATM consultant, FR Franco Gasperoni, ATM consultant Gilles Gawinowski, EUROCONTROL-EEC, FR Martin Michlmayr, University of Cambridge, UK Delphine Prieur, INRIA, FR Sip Swierstra, EUROCONTROL, BE Burkhardt Von Erlach, EEC, FR</p>	
<u>Date</u> 11 / 2006	<u>Pages</u> XII + 101	<u>Project</u> EEC-INO-OSIFE	<u>Period</u> 2005 to 2006
Distribution Statement: (a) Controlled by: Marc BOURGOIS, Deputy Mgr Innovative Research Department (b) Special Limitations: None			
Descriptors (keywords): Software, Open Source, CeCILL, COTS, FLOSS, GPL, IPR, OMG, OSIFE, OSS, SSS, W3C, ATM, ACAS, ADS, ANSP, ARTAS, ASM, ASTERIX, ATC, ATFM, ATM, BADA, CTAS, CFMU, CMB, CRCO, CHIPS, CNS, CTAS, DFS, EURET, GPS, FAST, FAA, FDPS, ILS, ICAO, NoGoZone, Mode-S, NATS, PSR, RTCA, RVSM, SSR, TCAS			
Abstract: <p>In the middle of 2005, the Innovative Research department of the EUROCONTROL Experimental Centre asked for the support of the CALIBRE project to organise a round table on the subject: "Potential of Open Source Software (OSS) in Air Traffic Management (ATM)".</p> <p>The objective of the round table was to increase awareness and to gather facts and arguments concerning four broad hypotheses about the possible implications of the OSS paradigm for the ATM. In order to reach this objective, the round table has been organised so as to foster cross-fertilization between two domains of expertise: the OSS expertise and the ATM expertise.</p> <p>Since this round table was a first open discussion about open source in ATM, it was decided to record all the presentations and discussions and to report it exhaustively for the benefit of further initiatives for the benefit of either ATM or OSS. This event has already triggered two similar initiatives in other industries than ATM.</p> <p>This report contains the transcript of all presentations and discussions, with an attempt to summarize some salient points. The website www.oss-in-atm.info contains a dynamic and audio format of this report.</p>			

ACKNOWLEDGEMENTS

by Jean-Luc Hardy and Marc Bourgois, local organizers

First of all, we would like to express our sincere thanks to the people of the CALIBRE-CALIBRATION community who have built the foundation for initiatives like this roundtable, by guiding the OSS movement for the benefit of specific industrial fields.

Our deep gratitude goes especially to Professor Brian Fitzgerald, CALIBRE project leader, who immediately and fully endorsed our project of a cross-fertilization workshop, and to Andrea Deverell, CALIBRE coordinator for the excellence of her commitment in co-organizing of the round table. Particular thank also to Franck Van Der Linden who kindly invited his colleague from PHILIPS to cast light on legal issues.

The panel of open source experts who participated in the roundtable consisted of: John O'Flaherty (CALIBRE, IE), Joseph Feller (University College Cork, IE), Franco Gasperoni (AdaCore Inc., FR), Delphine Prieur (INRIA, FR), Arnoud Engelfriet (PHILIPS, NL), and Martin Michlmayer (Debian project, University of Cambridge, UK).

We are very grateful to these academic and industry experts for the time spent in preparing the presentations, and for their open, creative and constructive contributions. The same goes for Jean-Dominique Frayssinoux, consultant in ATM, and for our EUROCONTROL colleagues who freely shared their insights and experiences: Gilles Gawinowski, Sip Swierstra, Carlos Garcia Avello, Burkhard von Erlach, and Garfield Dean. Many thanks to those who express their keen interest in the round table and to the public, a gathering of 15 professionals who followed the invitation from CALIBRE or the EUROCONTROL 4th Innovative Research Workshop. Their interventions have been most valuable.

Special thanks to Martina Jurgens for her help in the logistics for participants, to Herve Bechtel and Gilles Chedozeau for the voice recordings of the debates.

Finally, considering the proceedings, we want to thank John Seifarth and Colette Uytterhoeven (Words and Wires, BE) for transcripts of all recordings and the bright idea of initiating a web site to broadcast the complete proceedings. Since this web site was built in an open source spirit, we could enhance its structure and content to emphasize the rich outcome of the roundtable.

The photos of the sessions are the work of the sharp eye of John O'Flaherty, whose notes were a precious aid in writing summaries.

TABLE OF CONTENTS

ACKNOWLEDGEMENTS	V
OSS GLOSSARY	XI
ATM GLOSSARY	XII
1. INTRODUCTION	1
1.1. A PRIORI: A CROSS-FERTILIZATION PRINCIPLE	1
1.2. A POSTERIORI: PROCEEDINGS AND FOLLOW-UP	1
2. PROGRAM.....	5
2.1. PART 1: INTRODUCTION	5
2.2. PART 2: ATM PROJECTS	5
2.3. PART 3: IPR LEGAL ISSUES	6
2.4. PART 4: QUALITY AND SAFETY	6
2.5. PART 5: CONCLUSIONS.....	6
3. ABSTRACTS	7
3.1. EXPERIENCE REPORT ON WWW.OPENATC.ORG AND AUDIOLAN	7
3.2. WHITE BOX APPROACH FOR A TRAJECTORY PREDICTION TOOL	7
3.3. REVISITING CHIPS AS AN OPEN SOURCE PROJECT	7
3.4. COTS, FLOSS, AND MARKET FREEDOM IN SAFETY-CENTRIC INDUSTRIES.....	8
3.5. OSS AND IPR EVOLUTION IN EUROCONTROL	8
3.6. PERMISSIVE VS RESTRICTIVE OSS LICENSES.....	8
3.7. WORKING WITH OSS LICENSES IN PHILIPS	9
3.8. TCAS BEING OPEN SOURCE BEFORE THE TERM WAS COINED.....	9
3.9. QUALITY IMPROVEMENT AND RELEASE MANAGEMENT	9
3.10. OSS IN SECONDARY SOFTWARE SECTOR, VOICE OF INDUSTRY.....	10
4. PRESENTATION OF CALIBRE.....	11
4.1. OVERVIEW	11
4.2. DISCUSSION	12
4.3. SALIENT POINTS	12
5. PRESENTATION OF EUROCONTROL	13
5.1. OVERVIEW	13
5.2. SALIENT POINTS	13
6. HYPOTHESES ABOUT OSS IN ATM	15
6.1. OVERVIEW	15
6.2. SALIENT POINTS	17
7. MIGRATION TO OSS	19
7.1. OVERVIEW	19
7.2. DISCUSSION	22
7.3. SALIENT POINTS	22

8. EXPERIENCE REPORT ON OPENATC.ORG AND AUDIOLAN	23
8.1. OVERVIEW	23
8.2. DISCUSSION	25
8.3. SALIENT POINTS	27
9. WHITE BOX APPROACH FOR A TRAJECTORY PREDICTION TOOL	29
9.1. OVERVIEW	29
9.2. DISCUSSION	30
9.3. SALIENT POINTS	31
10. REVISITING CHIPS AS AN OPEN SOURCE PROJECT	33
10.1. OVERVIEW	33
10.2. DISCUSSION	33
10.3. SALIENT POINTS	35
11. COTS, FLOSS, AND MARKET FREEDOM IN SAFETY-CENTRIC INDUSTRIES	37
11.1. OVERVIEW	37
11.2. DISCUSSION	39
11.3. SALIENT POINTS	41
12. OSS AND IPR EVOLUTION IN EUROCONTROL	43
12.1. OVERVIEW	43
12.2. DISCUSSION	47
12.3. SALIENT POINTS	48
13. PERMISSIVE VS RESTRICTIVE OSS LICENSES	49
13.1. OVERVIEW	49
13.2. DISCUSSION	51
13.3. SALIENT POINTS	51
14. WORKING WITH OSS LICENSES IN PHILIPS	53
14.1. OVERVIEW	53
14.2. DISCUSSION	56
14.3. SALIENT POINTS	57
15. TCAS BEING OPEN SOURCE BEFORE THE TERM WAS COINED	59
15.1. OVERVIEW	59
15.2. DISCUSSION	62
15.3. SALIENT POINTS	63
16. QUALITY IMPROVEMENT AND RELEASE MANAGEMENT	65
16.1. OVERVIEW	65
16.2. DISCUSSION	67
16.3. SALIENT POINTS	67
17. OSS IN SECONDARY SOFTWARE SECTOR, VOICE OF INDUSTRY	69
17.1. OVERVIEW	69
17.2. DISCUSSION	71
17.3. SALIENT POINTS	73

18. PRELIMINARY DEBRIEFING: LESSONS LEARNED	75
18.1. OSS IS A REALITY FOR EVERY BUSINESS	75
18.2. LICENSES ARE NOT THE BARRIER.....	76
18.3. COMMUNITIES THAT ARE SUCCESSFUL.....	76
18.4. CREATING VALUE WITH OSS	77
18.5. NO OSS EXPERIENCE IN SAFETY-CRITICAL DOMAINS	78
19. ROUNDTABLE PROCESS.....	79
20. SHORT BIOGRAPHIES.....	81
21. PARTICIPANTS.....	87
FRENCH TRANSLATION (RESUMÉ EN LANGUE FRANÇAISE).....	89

ATM GLOSSARY

Abbreviation	De-Code
ACAS	A irborne C ollision A voidance S ystem (tool)
ADS	A utomatic D ependant S urveillance (tool)
ANSP	A ir N avigation S ervice P rovider (service)
ARTAS	A TC R adar T racker A nd S erver (tool)
ASM	A ir S pace M anagement (domain)
ASTERIX	EUROCONTROL format for radar data (standard)
ATC	A ir T raffic C ontrol (domain)
ATFM	A ir T raffic F low M anagement (domain)
ATM	A ir T raffic M anagement (domain)
BADA	EUROCONTROL B ase of A ircraft D Ata (tool)
CTAS	C enter T RACON A utomation S ystem (tool)
CFMU	EUROCONTROL C entral F low M anagement U nit (service)
CMB	Combined P SR + S SR radar (tool)
CRCO	EUROCONTROL C entral R oute C harges O ffice (service)
CHIPS	C ommercial H ighly I nteractive P roblem S olver (tool)
CNS	C ommunication, N avigation, and S urveillance (domain)
DA	C TAS D escent A dvisor (tool)
D2	C TAS D irect T o (tool)
DFS	A NSP in Germany (company)
ESCAPE	A TC simulator developed by EUROCONTROL-EEC (tool)
EURET	C TAS C onflict detection tool (tool)
GPS	G lobal P ositioning S ystem (tool)
FAST	C TAS F inal A pproach S pacing T ool (tool)
FAA	U SA F ederal A viation A uthority (organization)
FDPS	F light D ata P rocessing S ystem (tool)
ILS	I nstrument L anding S ystem (tool)
ICAO	I nternational C ivil A viation O rganization (organization)
No Go Zone	Area with high probability of conflict (tool)
Mode-S	Radar protocol for interrogation of a specific aircraft (standard)
NATS	A NSP in England (company)
PSR	P rimarily S urveillance R adar (tool)
RAMS	F ast-time simulator developed by EUROCONTROL-EEC (tool)
RTCA	R adio T echnical C ommission for A eronautics (organization)
RVSM	R educed V ertical S eparation M inimum (project)
SSR	S econdary S urveillance R adar (tool)
TCAS	T raffic alert and C ollision A voidance S ystem (tool)
TP	aircraft T rajectory P rediction (tool)

1. INTRODUCTION

by [Jean-Luc Hardy](#), EUROCONTROL

ATM is part of the Secondary Software Sector (SSS), i.e. the industries where software is considered as an enabler or a component of a product or service, but not as the final product in itself. Presently, there is no model and no published examples for the adoption of OSS in SSS. Therefore, new initiatives must be taken towards such adoption. The process of these initiatives must be described explained and analysed, in order to allow refinements, dissemination, and innovative modelling.

A round table was organized at the EUROCONTROL Experimental Centre on 7th Dec. 2005 in order to discuss the adoption of OSS in ATM. The present introduction reports about two particular aspects of the process of the round table: a cross-fertilization principle and extensive proceedings to allow follow-up initiatives.

1.1. A PRIORI: A CROSS-FERTILIZATION PRINCIPLE

The [brochure](#) prepared by organizers to introduce the roundtable contained the following explanations.

In the middle of 2005, the Innovative Research department of the EUROCONTROL Experimental Centre asked for the support of the CALIBRE project to organise a round table on the subject: "Potential of Open Source Software (OSS) in Air Traffic Management (ATM)". EUROCONTROL's objective is to increase awareness and to gather facts and arguments concerning four broad hypotheses about the possible implications of the OSS paradigm for the ATM (in particular for Air Traffic Control).

In order to reach this objective, the round table has been organised so as to foster cross-fertilization between two domains of expertise: the OSS expertise and the ATM expertise.

Speakers were invited to consider the questions presented in the introduction to the book [Perspectives in Free and Open Source Software](#)¹.

The program has been arranged to alternate presentations about ATM initiatives and presentations from the CALIBRE group concerning OSS.

The EUROCONTROL initiatives are pioneer actions towards the introduction of OSS in ATM projects. The OSS presentations consider ATM as a possible case study for the introduction of OSS in the Secondary Software Sector.

1.2. A POSTERIORI: PROCEEDINGS AND FOLLOW-UP

This roundtable was a première, for CALIBRE and for EUROCONTROL. For CALIBRE, it was the first in-house workshop of the industrial forum CALIBRATION. For EUROCONTROL, it was probably the first time that the issue of OSS in ATM was thoroughly discussed by experts invited from both domains. Normally, these people have very little chance to meet and to share their insights: ATM specialists are involved in ATM projects and have no time to visit OSS conferences to improve their OSS insights; OSS specialists have not much time to consider the specific issue of OSS in ATM, despite the fact that they can immediately provide relevant feedback when this issue is raised.

¹ Feller, J., Fitzgerald, B., Hissam, S., Lakhani, K. (eds.), 2005, Perspectives on Free and Open Source Software (MIT Press, USA-Cambridge, ISBN 0-262-06246-1).

The quality of the presentations, the number of reported interventions (216) and the value of the arguments that were discussed are evidence that the a priori objectives for this event have been met, both for CALIBRE and EUROCONTROL.

In terms of follow-up for the ATM community, there is a reference corpus of arguments that people can visit and revisit to increase drastically their awareness and build their own opinion about the potential of OSS in ATM. Some participants to the roundtable are now considering practical consequences of the roundtable for specific projects, but this may take time and will depend on managerial decisions and negotiation with the industrial partners in the ATM industry.

The ATM case is now mentioned by the participating OSS experts in conferences and workshops. Such a case study is of mutual benefit for both the OSS and the ATM field.

In terms of follow-up for the OSS community, this roundtable has triggered similar initiatives in other industrial domains. On 4th May 2006, at Madrid, CALIBRE organized a second cross-fertilization workshop with Vodafone. Another one is planned at PHILIPS (19th Sept 2006, The Netherlands).

This event has also triggered interest from academic OSS specialists who questioned to what extent this event could be used as a model and repeated or extrapolated to other industrial context. The effectiveness of such events depends on actions taken before, during and also after the event:

- The choice of the guest speakers, a careful preparation of the roundtable program, and the room arrangement should stimulate interactions between the industrial experts presenting their projects or experiences and the OSS experts providing feedback. The "roundtable" formula emphasizes the value of feedback, discussions and brainstorming which are necessary especially at the early stage of a new phenomenon, like the introduction of OSS in an industrial field.
- The special role of the moderator in such a roundtable is to keep the debate focalized on cross-fertilization issues.
- The long term benefit is probably dependent on the quality of the proceedings. The a posteriori objective concerning the roundtable is to disseminate these proceedings. In addition to the EEC report, the website www.oss-in-atm.info provides an HTML copy of the proceedings together with the presentation files and the audio files of the complete recordings. Some innovative structural (not domain specific) features were introduced in the format of this website to emphasize the value of all interactions resulting from the cross-fertilization process².

Here are a few reasons why much emphasis has been put on the proceedings.

- The comprehensive proceedings should underline the importance of this initial event for the introduction of OSS in ATM.
- The proceedings should help disseminate the ideas emerging from the cross-fertilization process. The complete proceedings of the roundtable, including filtered voice recordings³, will allow experts not present at the roundtable to study the potential of OSS in ATM, as if they were silently present at the roundtable. The linked interventions of each participant might trigger further contacts between experts.

² Two pieces of JavaScript code created to increase the security of this website were put on www.sourceforge.net under the project names No Spam Mailto and Disable Right Click.

³ Thousands of noises and hesitations in the original audio recordings have been filtered. The length of the audio recordings has been reduced by more or less 20%. This filtering process explains the delay of the production of the proceedings. Now the filtered audio recordings can be listened as if the speakers were reading their text in a dynamic way. However, for some parts, the sound quality is low, because backup recordings had to be used.

- For those who were present, the number of interactions was important during and after most of the 15 presentations: 216 interventions are reported in these proceedings. The roundtable covered a wide range of debates and it is not possible to remember all facts and arguments after just hearing them once. This website allows browsing again to the debates in every detail.
- There will not be a second chance to have a first roundtable. The motivation shown by participants on such initial event will not necessarily be present on a second edition of the same event. Detailed transcripts and voice recordings have the advantage of showing participants interests, convictions, emotions, fears, emphasis and creative ideas. These behavioural features could motivate people more than text-only proceedings. The complete proceedings of the first edition are essential also to keep track of all initial and divergent ideas before any consensus could take place.
- Unlike other types of events, it will not be easy to repeat such a roundtable, due to the lack of availability of experts in the intersection of the two domains. If a second roundtable were to be organized in the same industrial domain (ATM), the proceedings of the first one would be background information for preparing the program of this second edition. The proceedings would help build the second roundtable while avoiding repetitive ideas and useless meetings. Reviewing the roundtable could be beneficial as the detailed proceedings are meant to be a building block for further initiatives and decisions about the introduction of OSS in ATM.

2. PROGRAMME

2.1. PART 1: INTRODUCTION

9:00 Opening

- Agenda and participants, by Marc Bourgois.
- Presentation of CALIBRE, by John O'Flaherty.
- Presentation of EUROCONTROL, by Jean-Luc Hardy.

9:30 Introduction to ATM (Air Traffic Management)

- Hypotheses about OSS in ATM, by Marc Bourgois, EUROCONTROL.
- Discussion.

9:50 Introduction to OSS (Open Source Software)

- Migration to OSS,
by Joseph Feller, University College Cork.
- Discussion.

10:30 Coffee Break

2.2. PART 2: ATM PROJECTS

11:00 ATM: hypothesis = Public Service

- Experience Report on www.OpenATC.org and AudioLan
by Gilles Gawinowski, EUROCONTROL.
- Discussion.

11:30 ATM: hypothesis = Harmonization

- White Box approach for a Trajectory Prediction Tool,
by Sip Swierstra and Carlos Garcia Avello, EUROCONTROL.
- Discussion.

12:00 ATM: hypothesis = Business Model

- Revisiting CHIPS as an Open Source Project,
by Jean-Dominique Frayssinoux, ATM consultant.
- Discussion.

12:30 OSS feedback

- AdaCore Business Model and ATM,
by Franco Gasperoni, AdaCore, Inc.
- Discussion.

13:00 Lunch

2.3. PART 3: IPR LEGAL ISSUES

14:00 ATM: hypothesis = Public Service

- OSS and IPR Evolution in EUROCONTROL, by Burkhard von Erlach, EUROCONTROL.

14:20 OSS feedback

- Permissive vs. Restrictive OSS Licenses, by Delphine Prieur, INRIA.
- Working with OSS Licenses in PHILIPS, by Arnoud Engelfriet, PHILIPS.
- Discussion.

15:30 Coffee Break

2.4. PART 4: QUALITY AND SAFETY

16:00 ATM: hypothesis = Quality, Safety

- TCAS being Open Source before the term was coined, by Garfield Dean, EUROCONTROL.

16:30 OSS feedback

- Quality Improvement and Release Management, by Martin Michlmayr, University of Cambridge.
- Discussion.

2.5. PART 5: CONCLUSIONS

17:00

- OSS in Secondary Software Sector, Voice of Industry, by John O'Flaherty, CALIBRE.
- Discussion and conclusions.

17:30 Close

3. ABSTRACTS

3.1. EXPERIENCE REPORT ON WWW.OPENATC.ORG AND AUDIOLAN

by Gilles Gawinowski, EUROCONTROL

Abstract. AudioLAN & OpenATC represents several OSS initiatives. Lessons learnt seem to demonstrate that the condition and interest to support this approach are not met:

- ATM research mass is too small.
- The process to transfer research into product and the customer-oriented approach are not adequate with an OSS approach.
- The core-business re-focus and outsourcing management decreased drastically software activities.
- Not a driver to resolve ATM research issues.

3.2. WHITE BOX APPROACH FOR A TRAJECTORY PREDICTION TOOL

by Carlos Garcia Avello and Sip Swierstra, EUROCONTROL

Abstract. Many critical and costly elements in the ATM system development process could well be shared without affecting the technical excellence of individual industries and their competitive advantages. A community approach to the development, validation and verification of such common elements is the most efficient way forward. The process of white boxing a TP aims at making it and the know-how that has been applied and accumulated during the development easily accessible to the ATM community. Due to these constraints the TP distribution will not be suitable for operational real-time applications. However, if the product is provided as an OSS, any of its components could be used within existing applications. Behind the objective of providing the TP users with the possibility to analyze the insides and decide on the trade offs to be accepted, the provision of a white box TP aims at paving the way to the development of OSS widely used TP components. This will ensure the interoperability across ground TP clients and will enable the synchronisation of airborne and ground TPs.

Most important point for the future of OSS in ATM. Probably the most important benefit of OSS in ATM is the possibility to use that vector for ensuring ATM systems interoperability. The TP being one of the underlying key functions of the future Trajectory Based ATM concept, providing OSS TP elements, will facilitate systems interoperability and air and ground trajectory consistency.

3.3. REVISITING CHIPS AS AN OPEN SOURCE PROJECT

by Jean-Dominique Frayssinoux, ATM consultant

Abstract. First part of this presentation gives objectives and advantages of OSS in ATM environment. Second part of this presentation describes an OSS in the near future: CHIPS program. This software is a tool that provides assistance to the controllers to enable efficient resolution of conflicts between aircrafts.

Most important point for the future of OSS in ATM. OSS is a good opportunity to present useful software and to show ATM expertise. Distributing OSS on a web site provides useful contacts with ATM organisations and ATM companies in the whole world.

3.4. COTS, FLOSS, AND MARKET FREEDOM IN SAFETY-CENTRIC INDUSTRIES

by Franco Gasperoni, AdaCore, Inc

Abstract. This presentation discusses the relationship between commercial off-the-shelf software (COTS) and Freely Licensed Open Source Software (FLOSS) from a purely business perspective. The emphasis is on safety-centric industries such as aerospace (the full executive summary is at the beginning of the paper).

Most important point for the future of OSS in ATM. Our works shows that all other things being equal a freely licensed COTS is always better, for the customer, than a restrictively licensed COTS because the lack of vendor lock-in for software changes, support, and certification material aligns the COTS vendor interests with the customer's around high-quality products and services at competitive prices. Put it another way, to the question: Should the ATM industry lobby with its vendor(s) so that, when it is of value, the vendor(s) make their COTS available to them with a FLOSS-like license? The answer is a resounding yes.

3.5. OSS AND IPR EVOLUTION IN EUROCONTROL

by Burkhard von Erlach, EUROCONTROL

Abstract. OSS is fairly new to ATM and EUROCONTROL. This presentation expands on some terms which are linked to some extent to OSS. It further explains the current policy of EUROCONTROL and highlights why EUROCONTROL as an international organisation needs to be cautious and is not as free as industry. OSS has not really taken up in ATM so far for various reasons. The author mentions a few of them and also lists a few conditions under which OSS might become more successful in ATM and wonders if this could not be an activity for EUROCONTROL.

3.6. PERMISSIVE VS RESTRICTIVE OSS LICENSES

by Delphine Prieur, INRIA

Abstract. The legal aspects of protecting software include moral rights and economic rights. A license is a contract that confers more or less rights to licensees. It is a regulation tool. Copyleft is not the antithesis of copyright. Counterfeiting is the violation of at least one of the rights attributed to a piece of software. Three major licenses are shortly presented: GPL, CeCILL and the new BSD. It is important to point out that the legal framework has not yet been tested and that there are compatibility problems between OSS licenses.

Most important point for the future of OSS in ATM. To choose a license, the question is not to consider whether it is permissive or not, but to ask relevant questions about the intentions for the developments (who will make the developments? what will be the status of the contributions?), and for the distribution (to whom and why?), keeping in mind that your intentions may change: it is better to use a flexible model.

3.7. WORKING WITH OSS LICENSES IN PHILIPS

by Arnoud Engelfriet, PHILIPS

Abstract. PHILIPS Electronics has used combinations of open source and closed, proprietary software for several years. This presents various challenges, such as how to safely combine the two, how to handle open source distribution requirements, and other things. Engelfriet will use PHILIPS' ABISS hard-disk scheduler (<http://abiss.sourceforge.net/>), which PHILIPS contributed to the Linux kernel, as a case study to show best practices for managing mixed-IP product development. For various reasons, PHILIPS wanted to keep certain parts of ABISS closed, and so the company developed a methodology to evaluate where this would be most appropriate and how this would be possible.

3.8. TCAS BEING OPEN SOURCE BEFORE THE TERM WAS COINED

by Garfield Dean, EUROCONTROL

Abstract. This presentation draws an analogy between the development of the TCAS (Traffic-alert and Collision Avoidance System) algorithms, and OSS. First, the TCAS system is very briefly described. Next a brief history of the development of the system is provided. In particular, it describes the roles played by two standardization bodies: the International Civil Aviation Authority (ICAO) and the Radio Technical Commission for Aeronautics (RTCA). In the main body of the presentation, the development of TCAS and OSS are compared and contrasted. Finally, the author speculates about where OSS might or might not be applied in Air Traffic Management (ATM).

Most important points for the future of OSS in ATM:

- Who will be responsible for any OSS development in ATM, especially when there is an accident like Überlingen?
- A related issue is whether we can trust OSS for safety critical systems. There is a risk of creating something that wants to be all things to all men (safe in all circumstances), but satisfies no-one (because it is unusually cumbersome).

3.9. QUALITY IMPROVEMENT AND RELEASE MANAGEMENT

by Martin Michlmayr, University of Cambridge

Abstract. It is sometimes argued that free software and open source projects are generally of higher quality than proprietary software. While there is good evidence that high amounts of peer review contribute to quality, there are also a large number of issues which have to be addressed. Development in free software projects is carried out in a distributed fashion and commonly performed by volunteers, which leads to unique challenges that have to be faced. This presentation will give an overview of common quality issues found in both large and small free software projects and then focus on release management, one problematic area in many free software projects.

3.10. OSS IN SECONDARY SOFTWARE SECTOR, VOICE OF INDUSTRY

by John O' Flaherty, CALIBRE

Abstract. This presentation addresses the OSS phenomenon from an industry perspective and reveals a number of complexities surrounding the role of OSS in the Secondary Software Sector (SSS). It presents the research results of an international workshop which was hosted with the explicit intention of extracting the voice of key industrial stakeholders. The data was gathered and analysed using a qualitative approach which revealed the key strengths and weaknesses of OSS from an industrial perspective. This formed the foundations for developing a framework describing the emerging commercial incarnation of OSS (we refer to this as Open Source Software, Inc.) The paper concludes that the European secondary software sector recognise the benefits of leveraging OSS but are aware of key issues pertinent to such an end.

4. PRESENTATION OF CALIBRE

by [John O'Flaherty](#), CALIBRE

4.1. OVERVIEW

The CALIBRE project (Co-ordination Action for Libre Software Engineering for Open Development Platforms in Software & Services) is a two-year coordination action within the 6th framework of the EU Information Society Technologies Program. The term "libre" in our name is intentional to emphasize that free software is free as in freedom, not just free of cost. We'll use the term OSS with its libre meaning. CALIBRE has 12 academic and research partners within Europe, and one industrial partner from China -- China Soft.

In the research part of our project, we have identified three scientific pillars key to the future of software development: OSS, Agile Methods, and Distributed Development.

Open source is an interesting paradigm, since it addresses the so-called "software crisis" of cost, delivery, and quality. Europe is quite active in open source. We should take advantage of this momentum. Most focus today has been with the governments and public authorities, but here we're looking at the secondary software sector, i.e. the sector where software is not the primary output. Open source is a complex phenomenon, and requires a multidisciplinary perspective. The CALIBRE consortium's multi-disciplinary leaders provide critical mass.

We have three objectives. The first is to integrate and coordinate libre (OSS) research and practice. Secondly, we want to foster the effective transfer of the lessons of open source between industry and academia. Finally, we want to establish a European industry forum, called CALIBRATION, to give voice to what European industry wants in open source, particularly to the Commission, of what policy initiatives are required.

Our three workshop packages are on open source, distributed development, and agile methods. Our two key outputs are the CALIBRATE industry forum, and a definition of a new paradigm, a new roadmap for software development. Our main tools are workshops and conferences, like today's roundtable.

To describe the expected impacts of CALIBRE, we can quote Alan Kay, "It's easier to create the future than predict it." For you in the ATM sector as well, while you can't predict the future, if you want it to happen, you can make it happen. We want to move from an ideological approach to open source towards a much more pragmatic and realistic research roadmap and agenda. We want to put open source on the agendas of sectors such as ATM. We hope to identify best practice business models. We will give back to the community more effective and coherent practice-informed research. Finally, we want to come up with a coherent vision or paradigm integrating what we see as the three pillars of software engineering: OSS, Agile Methods, and Distributed Development approaches.

4.2. DISCUSSION

from 6'25" to 10'30" (4'05").

G. Gawinowski: *What are the main objectives of CALIBRE?*

J. O'Flaherty: *CALIBRE is a coordination action. Its main objectives are not to create new technology, but to coordinate open source efforts in research and industry, and put open source on the public policy agenda. Explicit goals are:*

1. *To integrate and coordinate libre (OSS) research and practice - to ensure that the phenomenon flourishes and delivers to its true potential, especially for the European 'secondary' software sector (automotive, telecom etc) where Europe has particular strengths.*
2. *To foster the effective transfer of the many useful lessons from Libre (OSS) - to facilitate the next generation of software engineering methods and tools.*
3. *To establish a European industry OSS research policy forum - CALIBRATION.*

J-L. Hardy: *When EUROCONTROL started investigating the research and literature in the OSS domain, they needed to meet with specialists in the domain, and fortuitously encountered CALIBRE. CALIBRE's mission to facilitate the exchange of information between universities and different industrial sectors appeared to be a good partner in setting up a meeting to share experience.*

J. Seifarth: *Can you define the Secondary Software Sector in more detail?*

J. O'Flaherty: *The Secondary Software Sector (SSS) is EU-jargon for those industries where software is not the primary output, but where software is vitally important to the sector. Another useful way of looking at it is that SSS serves vertical markets, versus the horizontal markets of primary software sectors, like office software or database management software.*

The view at the EU is that the US dominates the primary software sector, while Europe is quite strong in the Secondary Software Sector. The USA has won the primary software war – but Europe could still prevail in the SSS. Is there some way to leverage these new software engineering paradigms, open source, agile methods, and distributed development to enhance our European position in the SSS?

Examples of SSS include ATM, automotive, telecommunications, and banking. Europe is strong in these sectors, but if we don't coordinate these efforts, we could fall behind to the Americans there as well.

J. Seifarth: *But this raises the questions of keeping development secret because they are assets to the companies that developed them.*

J. O'Flaherty: *Yes, these are issues that have to be addressed. At one level, open source is very straightforward technology. But further investigation raises a number of issues, like intellectual property rights and software patents. It is actually a very complex area, and it has major strategic implications which hopefully, we will explore today*

4.3. SALIENT POINTS

- We have identified three scientific pillars key to the future of software development: OSS, Agile Methods, and Distributed Development.
- Open source addresses the so-called "software crisis" of cost, delivery, and quality.
- "It's easier to create the future than predict it." (Alan Kay).
- The USA has won the primary software war, but Europe could still prevail in the Secondary Software Sector (SSS).

5. PRESENTATION OF EUROCONTROL

by [Jean-Luc Hardy](#), EUROCONTROL

5.1. OVERVIEW

Air Traffic Management is a rather catch-all term that actually covers three fairly distinct domains. The first is Air Traffic Control (ATC): surveillance to avoid collisions. ATFM is Air Traffic Flow Management, where airport capacities are factored in to avoid delays and planes parked in holding patterns above the destination airport. Finally, Air Space Management (ASM) involves optimum organization of the air space.

In order to support this ATM activity, there is a communication, navigation, and surveillance infrastructure (CNS). Communication involves communication between the pilot and the controller, or between controllers of different ATC centres. Navigation tools help an aircraft follow the proper route, and include beacons and ILS at airports, as well as satellites like GPS and soon Galileo. Surveillance can be active or passive. Passive surveillance uses radar, and GPS technologies have consistently improved active reporting of position.

EUROCONTROL has been involved in all these activities since the '60s. The first convention established the Maastricht control centre. In the '80s, flight delays became bothersome enough that ATFM grew in importance, and a big EUROCONTROL unit in Brussels was devoted to flight planning. More recently, ASM has been the response to rising air traffic, with the Reduced Vertical Separation Minimum (RVSM) project. All these activities require a lot of coordination between the different Member States, and that is the role of EUROCONTROL.

EUROCONTROL has 35 Member States: all the EU Member States except the Baltic states of Latvia, Lithuania, and Estonia, plus 13 other non-EU countries, including for example, Turkey and Switzerland.

A few words about TCAS, before the more in-depth talk that Garfield Dean will present: Traffic alert and Collision Avoidance System is an on-board, last-minute collision detection system. If the ground control system fails, TCAS is the last chance at avoiding a collision.

The main mission of EUROCONTROL is safety. There are other missions that EUROCONTROL handles as well: increasing the capacity of air traffic in general, improving efficiency, improving security (especially since 9/11), and last but not least, the environment. EUROCONTROL coordinates big, pan-European programs, does R&D especially in this building, training in Luxembourg, operational ATC in Maastricht and (sooner or later) in Budapest for Eastern European coverage. Flow management operates out of the Central Flow Management Unit (CFMU) in Brussels, with a backup near here.

5.2. SALIENT POINTS

- Air Traffic Management (ATM) groups three domains: Air Traffic Control (ATC), Air Traffic Flow Management (ATFM), and Air Space Management (ASM).
- A communication, navigation, and surveillance (CNS) infrastructure supports ATM activity.
- ATM activities require a lot of coordination between Member States, and that is the role of EUROCONTROL.
- TCAS is the last chance at avoiding a collision.
- The main mission of EUROCONTROL is safety. EUROCONTROL handles other missions at different centres in Europe.

6. HYPOTHESES ABOUT OSS IN ATM

by [Marc Bourgois](#) and [Jean-Luc Hardy](#), EUROCONTROL

6.1. OVERVIEW

The past year has been a year of discovery for the Open Source Implications For EUROCONTROL (OSIFE) project. We've been discovering contacts like CALIBRE, the principles of open source, and what has been done in EUROCONTROL and ATM before. We've come up with four hypotheses, which we will develop in more depth below. Observing the open source paradigm shift, the objective of the study is to determine how, if, and when OSS will impact the business of ATM.

The scope of OSIFE study is limited to ATM applications. We specifically exclude administrative and office software, as well as ATM systems infrastructure. Our focus is the applications layer for open source ATM software, i.e. the so-called Secondary Software Sector (SSS).

We started by reviewing the popular, technical, and scientific literature on open source. We defined the major hypotheses, and have been gathering facts and arguments on these hypotheses. Today's roundtable is the fruit of this preliminary work: we've gathered a series of open source and ATM specialists and actors in the area to analyze case studies. The prioritization of these efforts will be the result of today's meeting.

2005 has been a year of networking. A lack of awareness was discovered within EUROCONTROL. We observed that there were many people who had done something similar, not only within EUROCONTROL, but in industry as well. It's interesting to note that most attendees of this roundtable are from industry partners, not from research partners.

We've distilled a series of four hypotheses from our preliminary work, which should be validated or disproved by further work. We will now take a closer look at each.

The first hypothesis is perhaps the most important, given the role of EUROCONTROL: *Open source can be a facilitator for better harmonization and improve interoperability*. Traditionally, EUROCONTROL has tried to achieve interoperability between different ATM systems in the various member countries. These ATM systems are country-specific, with their own national suppliers and customized software. The task of EUROCONTROL has been to integrate the different systems.

Standardization attempts have met with limited success. Some successful standards include data formats for radar and coordinations between ATC centres, but they have been few: since EUROCONTROL has been established, a maximum of 6 standards have been created. Several have been very successful, but they have not solved the interoperability problem.

In the '90s, the European Harmonization Program tried another approach: EUROCONTROL developed the software, and gave it to all the partners. The idea was that this would improve harmonization, because everyone would be running the same software. Examples include ARTAS, a radar tracking system. EUROCONTROL stopped that approach, since we were pushing other suppliers out of the market and creating monopoly positions to the dismay of the European Commission.

The third approach involved multiple developers competing. Several smaller projects used this technique, with several development contracts concluded with different companies. Consolidation within the sector resulted in mergers between the former competitors, so we were back to square two, with a monopoly situation again.

A better approach to harmonization could be that we finance or facilitate open source ATM kernels that implement core functions. These standard kernels could be adopted throughout Europe, and maybe even other regions if they are successful. This would, of course, have an impact on business models, as we will discuss later. An example for this interoperability hypothesis will be discussed in a talk on a white box trajectory predictor, later today. EUROCONTROL is introducing another round of harmonization with the SESAR program. Perhaps OSS could be an alternative to market domination or hard core standardization, which is limited to interface definition, and which has not brought the full scope of benefits we had hoped for.

The second hypothesis concerns quality: *the quality of complex ATM software could be maintained, if not improved, through OSS development methods.* Quality covers two fundamental issues in our domain: safety and security. We must be sure that safety and security are at least maintained at current levels.

Over the year, we had hoped to learn about quality issues from research, but not a lot of research actually has been done, and from what is available, the issue is not black and white. It would be worthwhile to investigate open source quality influences, identify the variables involved, and determine their impact on ATM. For example, how does quality vary with the number of developers? The ATM community is small, but highly professional. How will a small, tightly focused community influence the quality spectrum?

There is a rather controversial argument that the safety criticality of ATM software is low, in contrast to the safety criticality of avionics. ATM controls the separation of aircraft. In the normal state of the system, the aircraft are separated. If the ATM system goes down for several minutes, the aircraft will still be separated when the system comes back up. The safety criticality situation for avionics is quite different -- real-time responses are required. Thus safety considerations are different in airborne and ground-based systems.

S. Swierstra: *As air traffic increases, there will be more integration with air traffic systems, and ground-based systems safety will become more important. More demands will be placed on the intelligence of ground systems, which will have safety impacts different from those we have today. Another safety argument is that the integration of airborne and ground-based systems is such that to add new equipment, one would have to certify ground-based systems like we certify avionics. However, certification of ground-based systems to meet end-to-end certification requirements would be massively expensive.*

Our hypotheses concerning *safety and security are defensive*: we want to make sure that by introducing OSS in ATM, we are not opening ourselves to new risks. We are essentially hoping that having more eyes on the source code can only improve it.

From a security perspective, we assume, and verify regularly with audits, that the ATM system is completely isolated from other networks. The ATM systems are integrated with each other, but are not integrated with wide-access public systems.

Our understanding of the *business model* is weak, and we hope that today's speakers will improve our understanding. The ATM application providing industry is quite small. The market is fragile, with big companies as major players, but which don't seem to make much revenue from such systems. The software is highly customized. Unlike aircraft cockpits, ATM controller positions are not standardized, with variations from country to country, and even between different software clients. This entails extra costs, but whether these costs are generating profits for the manufacturers is unclear.

We must address the issue of what business model makes sense if the ATM kernels are open source, and financed by all the parties, i.e. by EUROCONTROL. The revenue would come from what? The services and fringe products around it? It would be worthwhile to examine what the ATM industry would look like if the kernels are open source.

A major problem that has confronted EUROCONTROL is appropriation by suppliers. We pay for the development, over many years, and unavoidably create a monopoly, or pay for the development twice. We end up paying for the development, then paying for the product. We lose control of the software. It's not just lock-in, it's complete appropriation of the software.

EUROCONTROL is a *public service*. Everything we do should be in the interest of the public, in the interest of our member states. This goes beyond operationally usable software, to experimental and research software. At ATM research centres, software is developed for experimental purposes, and once the experiment is finished, this software languishes unused. There is outside interest in having access to this software. This would only have benefits for us. If we were to make it available, we would at least archive it and be aware that it exists. If we had a systematic way of opening up this software, it would not only benefit us, but our research and commercial partners as well. There is a counter-argument from our project managers: this internal research software does not necessarily have quality standards commensurate with EUROCONTROL's image.

An interesting parallel can be made with another release of intellectual property rights, in a domain far removed from ATM and open software. The BBC is releasing footage from their archives, and is on the verge of releasing even more material, including video and drama for which they have the IPR. Their argument is that since the public paid for this work, it should be available as a public service for the public to use creatively. They feel that there is an interest in the media community to create new productions, documentaries or films, based on existing material. For that to be possible, the existing material must be openly usable from an IPR point of view. The BBC has used their mandate as a public service to justify the release of the IPR.

This same public service argument could also be valid for EUROCONTROL, at least in the research community. Since the research was paid for by public money, there is no reason why these tools should not be placed at the disposition of other researchers.

P. Johnson: *EUROCONTROL should apply E. S. Raymond's law of "release often, release early". There's no problem in releasing with a 0.x release number, stating explicitly that it's an experimental prototype. The essential point is to release something!*

6.2. SALIENT POINTS

- We have identified four hypotheses:
 1. ATM harmonization will be facilitated.
 2. ATM quality will be improved.
 3. ATM industry will change but continue.
 4. ATM public service obligation will be met better.
- OSS could be the best approach to satisfy an important role of EUROCONTROL, which is to achieve interoperability and harmonization in ATM.
- There is not a lot of research available on the subject. How will a small, tightly focused community influence the quality spectrum?
- Our understanding of the business model is weak. What business model makes sense if the ATM kernels are open source?
- Since the research was paid for by public money, and EUROCONTROL has a public service mandate, there is no reason why these research tools should not be placed at the disposition of other researchers.

7. MIGRATION TO OSS

by [Joseph Feller](#), University College Cork

7.1. OVERVIEW

Communities collaborating to build software is as old as software itself. It was the dominant paradigm in the early days of computing. Richard Stallman formalized the concept in the mid '80s as "Free Software". Stallman enumerated four freedoms for software:

1. The freedom to run the software.
2. The freedom to study how it works, and adapt it to your needs.
3. The freedom to redistribute copies.
4. The freedom to improve the program, and distribute your improvements.

Stallman's approach is unabashedly ideological: his premise is that not only *can* software be free, but it *should* be free. Despite the linguistic ambiguities of the English word "free" which can imply both freedom as well as "no cost", Stallman choose to call his view on software "Free Software".

The business community, especially in the United States, was understandably nervous about these software freedoms and the term "Free Software". In the late '90s, the term "Open Source" software was coined on order to make the free software concept more palatable to business. This change in labelling seems to be quite successful, since we can observe widespread adoption of Open Source products and processes by industry.

Nonetheless, Richard Stallman's Free Software is actually the most useful definition for us, since the freedoms he has defined map quite nicely to EUROCONTROL's goals of harmonization, improvement of quality, public service, and creation of value.

There are numerous examples of successful Open Source products. Current OSS software available is quite mature, and is offered over a broad range of categories. Initial successes were mostly in server space, and some server software, like Apache or JBoss, are considered as "category killers" i.e. there are no significant competitors in their space, in terms of quality and popularity.

Interestingly, recent development has allowed OSS to make inroads onto the desktop. One often-heard criticism of the Open Source model is that it comes from developers, and is aimed at developers, and cannot be applied to end user needs in terms of usability or documentation for novice users. Now, however, there is a wide range of end-user oriented software available, capable of meeting most of the desktop-oriented needs of enterprises in place of proprietary software. Examples include the Firefox browser, the OpenOffice.Org office software suite, emails clients like Thunderbird, and the like.

Logically enough, the end user space where OSS really shines is developer tools. This is particularly useful for EUROCONTROL, because all the development software and the toolbox for building better software is there.

Open Source is defined by its terms of distribution, and that has several interesting implications. The actual development of software shifts from being a closed, proprietary effort that is released to a user group (which E. S. Raymond describes as "Cathedral-style" development) to an open community collaborating on the software. The other shift is that the source code, usually considered as the primary asset of a software firm, is now made into a commodity, and the software firm must define value using other things.

Firms can use this software to build better software in two ways. The first is that you can learn much, at both high and low levels, from the OSS out there. Stallman's "freedom to study" is essential to be able to learn from others' work.

Perhaps even more importantly, if the code wasn't there, the code asset of Open Source is still the community. This implies that by adopting Open Source, companies can access this community, and use it to further develop, document, debug and maintain their software.

One of the objections raised by critics of Open Source for EUROCONTROL is that as a very small group, would they gain value from this Open Source community? The answer is likely to be yes. In E. S. Raymond's words "*given enough eyeballs, all bugs are shallow*". Actually, in 2003 he amended that to say, "Given the *right* eyeballs, the most *important* bugs are shallow." What is important about the ATM community is not its size, but rather its expertise. You'd have the *right* eyeballs on the job. That may be enough in itself, to realize some of these advantages.

By shifting the software code from being a core protected asset to being a commodity, the software consumer is empowered. Vendor lock-in opportunities are mitigated. It significantly balances the power between the software producer and the software consumer.

Open Source changes the nature of software business models. To be an Open Source business, or to even be a hybrid proprietary/Open Source business, requires a fundamental shift from treating software as primarily a manufacturing/marketing activity to treating it as a service and knowledge-based activity, where the value of the software is in its use, not in its production.

Practically speaking, how does one adopt OSS? Some good practical insights can be found in the book *Open Source for the Enterprise* by Woods and Guliani (O'Reilly 2005). The core argument of the book is that using OSS means taking on the burden of overcoming the lack of productization.

The key value offer of the proprietary software industry is the package, bundling legal, support, and documentation services with software.

F. Gasperoni: *There is nothing inherently proprietary about bundling support, documentation, and software into a package, and some open source offerings are full products as well.*

OSS "in the wild", particularly for specialized research and experimental software such as in the context of EUROCONTROL, has a steep learning curve associated with overcoming this lack of productization.

A few things need to be considered in determining if a particular piece of software is right for a particular problem at EUROCONTROL, and if the open source process is right for developing and supporting a particular software development project.

First off, you must understand the problem. Again to refer to E. S. Raymond, a key driver for open source development is "*scratching their own itch*": developers work better on projects to meet their own needs, and they share this development with others with similar needs. EUROCONTROL needs to be able to understand its own itches, its own needs. As J. L. Hardy said, "EUROCONTROL doesn't produce much software, but lots of requirements." This is good, because you need clear enough requirements to be able to leverage the community.

Another important thing to ascertain is the relative importance of the system. This importance can range from experimental, research efforts to mission critical systems. Another facet is understanding your tolerance to risk, and how can success be measured? Is it successful if the software is useful, or is a success only if you rally an international community around it?

Understanding yourself as an organization, especially your skill levels. A series of skills are involved in adopting and producing OSS, and the usual IT department know-how is not necessarily enough: one skill requirement is the ability to skilfully engage with the community. This is a corollary of the lack of productization: when your support and development comes from the community instead of a commercial entity, it is vital to be able to engage the strength of that community for your purposes.

The final challenge is to understand the product. Finding good open source is not too hard, but finding the right software and right community for your needs may prove quite difficult. This can be a hidden cost for OSS: the cost of evaluating the available projects. Although there is no licensing fee, it might cost you almost as much to come to the confident answer that that product is right for you.

Evaluating the maturity of the software is not simple. Many aspects must be considered: product age, quality, momentum and popularity; usage costs to set up and support end users; ease of integration, modularity, standards, developer support. In any case, the most vital aspect remains the community. If you are going to build OSS, you have taken on the burden of building that community. If you are going to adopt OSS, you've taken on the responsibility of participating in that community. You have to answer questions about that community: who are their leaders? What are their goals? Is the culture of the community open? Is it growing and dynamic? How much commitment is there?

When you're building OSS, the roles are reversed: you need the skill set that you were looking for in the community. Your software may not necessarily be release-ready, but it should have an architecture that will allow it to grow with outside contributors. The culture of your community should support participation.

Is open source right for EUROCONTROL? Maybe. You can use these criteria to see:

- Can you match the right product or process to the right problem? More packaged open source solutions like office software could be a good area to get started. You could gain experience in participating in the community.
- If the cost of ownership is tightly related to licensing costs and vendor lock-in, then there can be a cost savings with OSS.
- If you can increase the value of the software by enjoying the freedom to modify the software, open source is an advantage.
- If you can build value by creating and participating in community, open source has much to offer, even if the software is equivalent to the proprietary counterpart.
- If you have an institutional need to share your software with the stakeholders who pay for it, then open source could be very appropriate.

7.2. DISCUSSION

from 27'20" to 31'27" (4'07" min)

R. Schreiner: *Open software is a great way of reducing risks because it gives the user more freedom, for example to fix problems by himself, but also a lot of responsibility. On the other hand, it can lead to a quite ineffective use of software. For example, some companies tend to try to do everything by themselves without enough expertise to do so. So if you use it wisely, it's great, but otherwise, it can make you lose a lot of time.*

J. Feller: *The quality and the management of risk improve with open source when there is the adequate know-how internally. You have to have the three pieces of the puzzle: the software itself, the licensing terms that gives you the freedom to improve it and engage with it, and the know-how. And if you don't know how, you can get involved with the community, because even if you can not fix the problem, you may be fully able to clearly articulate what you need fixed and chances are you are not the only one.*

R. Schreiner: *The problem with commercial software is that you must rely exclusively on the vendor for support and maintenance of the software, especially when the market is very small. If the vendor decides to discontinue the product, that's it! With an open source solution, if you're not satisfied with support, you can switch to another source of support. If you find a bug, that's important to you, you fix it yourself. You set your maintenance priorities, not the vendor.*

P. Johnson: *What about EUROCONTROL releasing software and continuing to work with the community, instead of finishing the project and going on to something else?*

S. Swierstra: *This is an excellent idea...*

7.3. SALIENT POINTS

- Richard Stallman defined four freedoms for software: freedom to run, study, redistribute, and improve software.
- Open source is a term coined to reassure business, which may be worried by the definition in terms of freedom.
- OSS has made wide inroads in server infrastructure, and more recently in desktop software as well.
- OSS moves software from being a core protected asset to being a commodity, empowering the software consumer.
- "Given the right eyeballs, the most important bugs are shallow."
- The key value offer of the proprietary software industry is the package, bundling legal, support, and documentation services with software. However, some open source offerings are full products as well.
- There can be a hidden cost for OSS: the cost of evaluating the available projects.
- If you are going to build OSS, you have taken on the burden of building that community. If you are going to adopt OSS, you've taken on the responsibility of participating in that community.
- If you have an institutional need to share your software with the stakeholders who pay for it, then open source could be very appropriate.

8. EXPERIENCE REPORT ON OPENATC.ORG AND AUDIOLAN

by [Gilles Gawinowski](#), EUROCONTROL

8.1. OVERVIEW

I started working with AudioLan around ten years ago, in 1995. AudioLan is a Voice Over IP (VOIP) system for Air and Ground Communication. At the time, voice over IP was a relatively unknown, pioneering technology, and our work was to see how this technology could be applied to our domain. Two OSS kernels were available: one from Berkeley and the other from INRIA. We used these two kernels to develop our radio and telephone system.

The software had to be customized for EUROCONTROL's needs and requirements, both in terms of features, and in terms of environment. For example, a supervision tool and a recording tool were needed. The architecture was reviewed in order to facilitate software maintenance and industrialization, with outsourcing objectives.

Other discussions address code quality in OSS, but our experience has been shades of grey, not black or white. Our reason for using these open source kernels is that it gave us free access to the efforts of academic research in the VOIP domain. The quality, architecture, and languages chosen were perhaps less appropriate for our mission and the industrialization requirements we had.

For us, it was a success. The system was deployed in 10 European sites, with about 500 positions. It was then proposed as freeware to industry and research centres, in order to disseminate our research. 10 licenses agreements were signed with industrial partners, which stipulated that we were giving this software for free, but that they should contribute back any modifications they make. The goal was to have a win-win mechanism.

The next step was industrialization. In this step, industrial enterprises re-engineered the industrial design, to deliver features according to user requirements. What we delivered as a white box became converted into a black box. Why did this happen? In part, because there is some ambiguity about our mission as a public research centre. When we develop interesting technology that will be used by our clients, we need to take into account issues like support, installation, maintenance, and continued development of the software. The role and mission of a public research centre is to make innovation and promote this innovation. It is not to become an industrial actor, we don't want to support, install, or maintain the innovation afterwards.

Thus industry is the key factor to promote and use the research results. However, in the process, the open source research project is irremediably lost. In our experience, each customer has completely re-engineered our software and turned it into a black box. The OSS philosophy is lost, and the product again becomes proprietary. The cost for clients remains the same, compared to former products, since the added value industry provides is quality process, documentation, support, and responsiveness. The advantage to the client is the access to the innovative technology resulting from the research effort -- in our case, clients can switch from traditional telephony to communications based on Internet protocols using VOIP.

We can now fast-forward to around 6 years ago, in 1999, when we tried to create a community of researchers on the OpenATC project. Two research centres collaborated to develop a research community in ATM through a forum. The objective was to stimulate Air Traffic Control research and development by providing an easy way to share resources. We wanted to create a community, with not only software researchers, but operational people as well. Above all, we wanted to stimulate the exchange of ideas about the future of Air Traffic Management systems.

We created a web site for free software. Initially, everybody was quite enthusiastic about making interesting software available on the site, but we rapidly observed that when the strategic value of the software was high, the software wasn't posted on the site as freeware. Specifically, some participants raised a distinction about "added-value software", and tried to use that to cut a commercial deal with industry. While open source is a nice philosophy, when we have more concrete constraints in terms of political and strategic issues, and especially when money is involved, the philosophy changes.

F. Gasperoni: *Gilles, could you define "freeware"? For me freeware is neither OSS nor free software. So I would like to hear your definition.*

G. Gawinowski: *I am not a specialist. What we observe is that there are a lot of definitions, and different perceptions and understanding about what it means. I will come back about the possible ambiguity between the objective of a research centre about software and the open source community.*

F. Gasperoni: *So the binary code was available as freeware without a license.*

G. Gawinowski: *There was no license and no constraints.*

F. Gasperoni: *If there is no license, you cannot use it. But anyway...*

G. Gawinowski: *The objective was to create a community, not open source for open source.*

OpenATC failed. Why did it fail? the main negative factor is the fact that the ATM research community is too small: probably less than 1000 researchers in this specific area.

Besides that, at EUROCONTROL, research is no longer the core business. There has been a drastic reduction of research centres over the last ten years and software development is now often outsourced. There are only two research centres with sufficient research mass left in Europe. Some former research centres have become more IT service providers, some have become private companies. This EUROCONTROL research centre has few software engineers compared to ten years ago; now more of the personnel are managerial or operational.

Another problem has been the "not invented here" syndrome. Opening up code opened up a lot of duplicated efforts, reinventing the wheel. Many participants preferred to promote their own software rather than improve somebody else's software.

As for ATM research in general, six or seven years ago, there were a series of European Commission project initiatives to get the various parties involved together to work on interoperability and plug and play. This was just a dream, though, because now, after the intervening years, we can see we failed totally.

Their idea was to standardize the middleware, the API, and the applications. They wanted to force the industry to standardize at least these two components. Now we can see that the industry has not been following the game. They have been involved in these projects to observe, maybe learn something, and receive grant money. But there is not a single industrial example of standardized or common ATM middleware. None of the industrial players use the standard API either. In terms of applications that we wanted to promote on OpenATC, there were a lot of duplicated efforts, but no real idea of a community working together to try to push one application shared by all.

Therefore, in terms of middleware, API, and applications, there were maybe some limited successes, but when we observe the situation today, we need to recognize that we failed.

The question remains: Is there a future for OSS in the ATM area? There are issues:

What are the interest and quality of an OSS product in terms of architecture, functions, and interface, as well as support and maintenance? There has been a drastic reduction in ATM research centres, there is no critical mass. Most IT here has been outsourced. In any case software is no longer a core business of EUROCONTROL.

Our perception is that often, OSS is just opportunistic. It's just a freeware to capture clients and then to become proprietary under the pretext of additional features, like Skype. Some OSS wants to capture clients just seduced by the "free" religion. My opinion is that rather than accepting OSS as religious faith, we should ask what should be the added value to try to push open source in our specific environment. Some companies make software free for commercial opportunity.

Our market is small, with just a few key players in the industry. At the risk of being provocative, in my opinion there is no sign of evidence for an OSS future in the ATM area.

Beyond religious dogma, what are the objectives of an OSS approach? At a minimum we have our performance requirements, which are to make future ATM systems in terms of traffic capacity, efficiency, security, and environmental considerations. To meet our research objectives, we try to create something useful, and make public, transparent, understandable, and available.

These objectives may look like OSS but maybe they have nothing to do with OSS.

In my opinion as a computer science guy, who has been involved in technological project and OSS initiatives, the ATM research problem is not a technological issue but an operational one. Our main problems today are not technical, but operational. That is why I do not see how OSS can answer properly our problems.

8.2. DISCUSSION

from 19'42" to 35:42" (16'00")

[F. Gasperoni](#): *Could you define "operational" in this context?*

[G. Gawinowski](#): *Let's look at the difference between avionics and ground systems. There are few different avionics designs, just two big manufacturers' standards. In avionics, the design is driven by the manufacturer, because the aircraft is a technical object.*

In our area, the situation is quite different. We have to deal with over 50 different ground systems in Europe, because the requirements are driven by the users. The ground system is essentially based on the human operator, with his human ability to manage the situation. Thus, while ATM systems initially appear as very technical systems, they actually are human systems.

This human path is essential. When we use the word "operational" we mean the modus operandi. How does this teamwork operate, how do these individual human operators work together? It's more a question of working methods rather than technological issues.

[S. Swierstra](#): *ATM hasn't changed much from the seventies. The controller still sits behind a screen that gives them the positions of the aircraft approximately 5 seconds ago. In the '70s, the screen could only display green labels. Now, modern colour screens can display the labels in green as well.*

J. Feller: *The failure of www.OpenATC was a failure of community, not technology. It looks like OpenATC turned into a place for "show-and-tell", just a place where people could show off their code. Since you said you were provocative, I will be provocative in return: I would argue that the basis of the failure lies in the final slide. Instead of building something useful and making it public and transparent, you need to build something open, public, and transparent, and use the community to build the useful something. If you have essentially closed teams going off, building things, and then saying, "look what we've done!" then open source will never work. Show your mistakes, that is how you are going to learn.*

G. Gawinowski: *There is not a critical mass of ATM researchers to create an open source community. There are perhaps 1000 ATM researchers throughout Europe. Of them, even fewer are software engineers.*

J. Seifarth: *Why was it not possible to mandate that the APIs be published and more standardized APIs be used, when the contract for industrialization is written?*

G. Gawinowski: *Our previous attempts resulted in a monopoly situation. Either we try to regulate the market, or we let the market organize itself. The latter is the current approach, and we can note that in Europe, there are only about three key industrial players. These competing players have no interest in standardizing the API. They try to get clients to customize the software for their proprietary systems.*

M. Bourgois: *Around seven years ago, there was a lot of pressure from the Commission and from EUROCONTROL to establish common middleware, so that individual components would be more modularized. Service providers would be able to recombine these modules from different vendors, thanks to the common middleware and APIs. EUROCONTROL invested quite a bit in that, and Bretigny was a leading player in that effort. We have an ongoing project for building a common architecture, whom we have not invited today. We may be underestimating the value of that effort, but there was no obvious way to link it with open source.*

J-P. Florent: *The situation has changed in the intervening years. Since future systems are developed at European level instead of national level, we can hope that things will change.*

P. Kappertz: *I think there is an important issue missing. We are discussing from a user point of view why OSS is relevant or useful. But the missing point is why should the industry support OSS? What is the interest for the industry to support the OSS concept? You are only discussing the situation where the research results are given to the industry, but the industry is also developing something and it has to be paid for.*

M. Bourgois: *Let's leave that question pending now, because we have other interventions today from an industry viewpoint.*

R. Schreiner: *Were the problems with the common architecture technical or political?*

G. Gawinowski: *I think it is not a technical problem.*

R. Schreiner: *If you had community problems with this middleware, the problems with the open source community will be worse. In our experience, many in the community surrounding an open source project are just greedy. Many open source tools are used in industry by organizations who not only don't contribute to the project, but who aren't even willing to contribute their experience with the software as a success story.*

M. Bourgois: *For middleware, it's clear that the idea hasn't taken off. For architecture, the verdict is still out.*

Why was the AudioLAN development work re-engineered by each participant? Why went wrong in that scheme?

G. Gawinowski: *To a large extent, simply to meet their individual needs, and partly just because they could tinker with it, again reinventing the wheel.*

The level of commercial and competitive constraints varies with the level of the software involved. For instance, in graphics libraries or operating systems, ATM can benefit from open source. As we approach the core business of ATM, the working methods and the tools, like conflict detection and resolution algorithms, are the core business of industrial entities.

M. Bourgois: *Maybe we have to back on this issue when we discuss about business model, because it does not make sense for business to reinvent just for the pleasure.*

P. Johnson: *We need to look separately at two areas: research area and base technology of ATM.*

P. Crebassa: *The level of issues derived from the competitive situation depends on the level of the software we address. If we get closer to the core business ATM, for example conflict resolution algorithms are the core business of industrials and this may be the limit of the ATM communities and the open source principles.*

G. Gawinowski: *Software work like development of the graphics libraries, which I think is interesting, useful, and was consistent with some strategies 10 years ago, is no longer relevant to EUROCONTROL's current core business focused short-term strategy. Given such a trend, it is may be less relevant to think about open source today.*

P. Crebassa: *It seems French ATM R&D strategy is maybe a bit different from the other in Europe. So far, we consider that in essential topics like HMI or graphics, we still need some innovative capabilities.*

8.3. SALIENT POINTS

- AudioLAN is a VOIP system adapted to ATM needs.
- AudioLAN used OSS not for code quality, which was found to be variable, but rather for the free access it offered to academic research efforts.
- The role and mission of a public research centre is to make innovation and promote this innovation. It is not to become an industrial actor.
- Industry is the key factor to promote and use the research results. However, in the process, the open source research project is irremediably lost.
- Industry turned OSS research into a proprietary black box.
- OpenATC was intended to create a community, with not only software researchers, but operational people as well.
- We created a web site for free software, but we rapidly observed that when the strategic value of the software was high, the software wasn't posted on the site as freeware.
- The ATM community is too small for critical OSS mass.
- Research is no longer a core business at EUROCONTROL.
- The interoperability initiatives of the past 7 years, based on common middleware and APIs, has failed to produce the desired results.
- Opinion: there is no sign of evidence for an OSS future in the ATM area.
- Opinion: the ATM research problem is not a technological issue but an operational one.
- While ATM systems initially appear as very technical systems, they actually are human systems. It's more a question of working methods rather than technological issues.
- Perhaps software work like development of the graphics libraries is no longer relevant to EUROCONTROL's current short-term strategy.

9. WHITE BOX APPROACH FOR A TRAJECTORY PREDICTION TOOL

by [Carlos Garcia Avello](#) and [Sip Swierstra](#), EUROCONTROL

9.1. OVERVIEW

The aircraft Trajectory Prediction tool (TP) is not an open source project, but it is a good example of a critical ATM element that could be shared. Sharing this tool would facilitate building a community around it. A community approach to development, validation, and verification could increase efficiency significantly. A White Box approach to the trajectory prediction tool will make the know-how applied and accumulated during the development process available to the ATM community.

A bit of background about trajectory prediction: TP is used practically everywhere in ATM systems. On the ground, it is used for preflight and real time flow management and for take off. When the aircraft is airborne, it is used in all flight data systems: for collision avoidance, guidance, and other flight tools. TP is used for multi-sector traffic, and for new tools dealing with planning and traffic separation. It is also used within airline operations centres for optimum city pair flight planning.

Future trends in ATM point toward trajectory-based ATM. Especially on the ground, the clients for this trajectory prediction have different requirements. Under the Common Trajectory Modelling Initiative, we've built a white box TP system which could be shown to clients, so that they use the white box requirements to determine the requirements for their final application they're building.

Currently, there are about 40 or 50 different systems on the ground, each with a different trajectory predictor. Airborne systems are limited by the few manufacturers who produce the systems, but there are different as well. The figures are probably even worse, since the trajectory predictor is quite often implemented within the application itself, so a single system running multiple applications can have several different TP implementations. This can lead to inconsistencies.

To address consistency, we still separate air systems from ground systems. But within the ground systems, the goal would be a single system, or at least consistent separate systems. Thus the same tool could be used both for planning and at the tactical level, and even for the new tools implementing trajectory-based ATM. However, as long as there are separate systems for air and ground, specific attention must address consistency between them.

We have attempted to define the generic structure of a trajectory predictor, in the air and on the ground. The input comes from a "flight script". This set of instructions is input into a software process, the TP engine, which interprets the script like a language. To actually compute the trajectory, it needs met data, aircraft performance data to be able to emulate the behaviour of the aircraft, and the TP function library. The output of the TP engine is a trajectory, which can be used as the input of other service routines. For example, a flight data processing system can use this as input to calculate when the aircraft will arrive in a given sector, and when it will exit.

One of the important issues we have identified is this flight script, which allows the trajectory prediction process to be decoupled from the application requirements. The script should not only specify what should be computed, but also how it should be computed. Depending on the application, various processing tradeoffs can be made. For example, for real-time flow management, many trajectories must be computed in a short period of time, for a long time in advance. However, lower accuracy is acceptable. In contrast, conflict probing requires higher accuracy. This is why a common aircraft intent description language needs to be developed, which can be used in the different applications.

The flight script must be capable of supporting different modes of operations. The new tools being developed have different needs. For example, in open loop applications, the trajectory is predicted, and the progress of the flight is compared to that prediction. On the other hand, in closed-loop systems, you apply a constraint to the trajectory prediction through the flight script interface, and run the TP until the constraint is met. You then generate advisories for the controller, to be followed in order to match the constraint.

From the white boxing process, we have identified components which could be open source. First off, the components including the preparation process, trajectory update process, and TP export process, as well as services, should be part of the proprietary application developed by the flight data processing vendor. By defining this intent description language, which can be the interface between the flight script and the trajectory engine, we have determined that the met library, TP function library, aircraft performance library and even the TP engine could be open source. This could offer large advantages in terms of interoperability.

The white boxing process started about 6 months ago. Currently, the aircraft and met functions are open sourced, and available as a library. We've also open sourced the API for decoupling the Aircraft Performance Model from the trajectory engine. The Aircraft Intent Description Language allows decoupling of the trajectory engine and the application.

We've developed a test harness to allow evaluation of multiple TP engines with the same reference data, validation strategy, and an agreed set of metrics. The major goal of this was to make the system understandable, and help other TP developers gather requirements. Thus, the client can test and determine that if they want a given level of accuracy at a given calculation speed, where they must put their TP engine development efforts, and what is the trade-off involved.

What are the benefits for us? By making one TP available publicly, we hope that will be a way of reaching the larger ATM community. Our emphasis remains improving interoperability, given the large number of different systems in Europe, and decreasing fragmentation, thus decreasing costs for the operators. Hopefully, as OSS, we will be able to benefit from improvements made by a wider community.

Some questions remain open. How can OSS in ATM systems improve efficiency, costs, and safety? Is OSS suitable for safety-critical applications? In the next 20 years, we hope to double air traffic. Humans alone will probably not be able to handle the increase, new tools are necessary. At that point, these tools will become safety-critical.

9.2. DISCUSSION

from 16'30" to 22'50" (6'20")

[P. Johnson](#): *Where can we download the trajectory predictor?*

[C. Garcia Avello](#): *It has not been released yet. But we would like to release it, not only as a TP engine, but as a test harness with a TP.*

[P. Johnson](#): *Please "release often, release early"! If this development would become more transparent, that would be great!"*

[M. Bourgois](#): *This is probably one of the important lessons from today: it is important to release early, even if it's a risk for the quality image.*

[F. Gasperoni](#): *We seem to be very fragmented in terms of ATM here in Europe. What's the story in the US?*

C. Garcia Avello: *They have a single system. The interest of the FAA participating in this TP study is that they are more worried about safety than about interoperability. They are dealing with the implementation of different tools and each of the tools has its own TP. If they have more than one tool in operation, the output of these tools could be at worst contradictory, and at least inconsistent, so they would be rejected by the controllers.*

F. Gasperoni: *In the US, is it a multi-supplier market?*

C. Garcia Avello: *There's a main system (CTAS), but there is a multiplicity of tools from different vendors. The core system aims at being consistent, but on that system, the FAA wants to plug in tools for conflict detection or planning (for example EURET, recently implemented by Lockheed Martin). These tools can not use the TP embedded in the flight data processing system (FDPS), often because it's too simplistic. So they come with their own TP, and there's a possibility of inconsistencies in the output.*

J. Feller: *You have very solidly identified the most opportune parts of the system. If you think of this as a business and not just an operational flow, the value-added activities of ATM are flow planning and collision detection and avoidance. Therefore trajectory prediction, related languages and meta-functions are just utilities.*

We can draw a parallel with the news media industry: there's a mark-up language called NEWSML initially developed internally by Reuters and adopted by the entire industry. Reuters footed the bill, not only for the language itself, but also to develop an open-source toolkit, which is similar to the engine that was just described. We can ask ourselves, why would an industry player do this? The news industry realized that to build value was not an internal issue but a network issue, and that they needed standardization to most effectively compete. Therefore they had to collaborate.

I see the same sort of process here: if we can agree on a data standard, on a processing standard, if we can exchange the utility software in a transparent manner, then vendors can go on adding value. It's a very hopeful endeavour.

C. Garcia Avello: *Yes, but we have to build a user community. Today, we have completely protected international IPR on software including the TP. There's no way, because of the IP rights, that you can make it work with a single TP. Without the strong community push, it will not happen.*

J. O'Flaherty: *When you say "we" do you mean EUROCONTROL or EUROCONTROL and the subcontractors?*

C. Garcia Avello: *It's EUROCONTROL and the FAA, because this work has been done specifically on the AP16.*

9.3. SALIENT POINTS

- TP is used practically everywhere in ATM systems.
- Future trends in ATM point toward trajectory-based ATM.
- A single system running multiple applications can have several different TP implementations. This can lead to inconsistencies.
- A flight script allows the trajectory prediction process to be decoupled from the application requirements.
- The white boxing process has identified components which could be open source.
- We've developed a test harness to allow evaluation of multiple TP engines with the same reference data, validation strategy, and an agreed set of metrics.
- Our emphasis remains improving interoperability, given the large number of different systems in Europe, and decreasing fragmentation, thus decreasing costs for the operators.
- Parallels can be drawn with the industry-specific standards of the news media industry.

10. REVISITING CHIPS AS AN OPEN SOURCE PROJECT

by [Jean-Dominique Frayssinoux](#), ATM consultant

10.1. OVERVIEW

The goal of putting CHIPS as OSS would be to distribute a set of ATM tools on Internet web sites. Users who want to download the software would fill in a form so that it would be easy to establish contacts. Similar to Red Hat, the next levels of support would not necessarily be free of charge. Once contact has been established with these ATM organizations, it would be possible to offer services, technical demonstrations, and show ATM expertise.

The CHIPS program was developed in EUROCONTROL Bretigny six years ago. It's a small program, between 10 and 20 thousand lines of C++ code. It's easy to install on a portable, it runs under Red Hat Linux. It can run as stand alone, using a file for its input data, or it can be connected to live radar data and ADS data.

Making CHIPS an open source project involves first releasing it under an open source license, with executable and source code available on a web site. Once contact is established, do presentations on CHIPS, and propose maintenance, support, and new development on the software.

The CHIPS Problem Solver is a tool that provides assistance to air controllers to enable aircraft conflict resolution. The Controller examines an air situation using Route window, Altitude window or Speed window. CHIPS displays no-go zones. The Controller evaluates one modification of one trajectory. CHIPS calculates new no-go zones after modification of this trajectory. The controller confirms new trajectory after evaluation of the trajectory modification.

CHIPS can accept data from multiple sources. It can handle live feeds from Primary, SSR, CMB radar stations, ADS sources, and flight plan information. It can handle a live connection with a tracker (ARTAS).

Further development could include improvements to current displays, adding a 3D display, comparing with ACAS information, and comparison with other conflict probe tools.

In conclusion:

- OSS is more interesting than shareware or freeware, since users can directly evaluate software and source code.
- Distributing OSS is a good opportunity to be in contact with ATM organizations or ATM companies.
- Technical propositions or technical demonstrations are easier after distributing OS software.

10.2. DISCUSSION

from 12'30" to 23'34" (11'04")

[M. Bourgois](#): *Jean-Dominique Frayssinoux had contacted EUROCONTROL a year ago to be able to put this software, a research tool initially developed for EUROCONTROL, on a web site, to show his ATM expertise and to make it available for all persons and companies interested. It serves our public role as well. Now I have a question. You propose to do extensions to the CHIPS software. Is that your intention to put them back into open source?*

[J-D. Frayssinoux](#): Yes, it is always possible to distribute new improvements to anyone interested in developing it. This would give them opportunities to show ATM expertise.

[J. Feller](#): Can you tell more about the license?

[J-D. Frayssinoux](#): In this case, the license is not yet finalized.

[J-P. Florent](#): How do you plan to market this software? Do you think that a web site is sufficient to bring people together around the project?

[J-D. Frayssinoux](#): We think so, but this is one of the first experiences of that type. The first step is to put the executable binary files and the source files on the Internet. The main advantage of the Internet is that anyone interested in the whole world could download the software.

[P. Crebassa](#): We have experience with this question, because the former CENA had made available a graphic tool and also the Digistrip, an advanced HMI tool based on a tactile screen. We experienced that we were contacted by many companies, not only companies specialized in ATC. According to these two experiences, it looks quite efficient. But it has to be pointed out that we were not a private company looking for benefits. We were more focused on community aspects.

[R. Schreiner](#): In our experience, OSS can be a wonderful marketing tool. Since we are a very little company, normally big companies will not talk to us. However, in the open source process, it is easier to get support contracts, and then the project grows and grows. OSS is a very good entry point to the big players.

[M. Bourgois](#): That's promising!

[O. Robert](#): CHIPS seems to be interesting, but since it is highly specialized, don't you think that it will be difficult to build a community around it? Just putting it on a web site is not enough.

[J. Feller](#): I agree that a community is not just a matter of building a dot com. However, I strongly disagree on the first comment: that because it is a sub-component of larger system, that it works against open source; it actually works in favour of it. In the early free software open source history, most early open-source successes were small components of bigger systems. I do not think small specialized functionalities work against you.

[M. Bourgois](#): The argument being that it is a very specialized tool. Could you recombine it to widen its use?

[J-D. Frayssinoux](#): It is too early to answer such a question. I must start first to distribute it.

[J. Seifarth](#): It has to have a trajectory prediction engine in it. ;-)

[J-D. Frayssinoux](#): Indeed, but the TP needs first of all a flight plan information.

[J-L. Hardy](#): In order to facilitate the communication, I would like to say that Ollivier Robert is one of the main actors of the FreeBSD project.

[O. Robert](#): Indeed, I am here not so much as a EUROCONTROL person, but mostly as a software guy. I have been playing with software for more than 15 years.

10.3. SALIENT POINTS

The CHIPS Problem Solver is a tool that provides assistance to air controllers to enable aircraft conflict resolution.

- The goal of putting CHIPS as OSS would be to distribute a set of ATM tools on Internet web sites.
- The CHIPS program was developed in EUROCONTROL Bretigny six years ago, and is between 10 and 20 thousand lines of C++ code.
- CHIPS can accept data from multiple sources, including PRI, SSR, CMB radar stations, ADS sources, and flight plan information, and live connections with a tracker (ARTAS).
- Distributing OSS is a good opportunity to be in contact with ATM organizations or ATM companies.
- Technical propositions or technical demonstrations are easier after distributing OS software.

11. COTS, FLOSS, AND MARKET FREEDOM IN SAFETY-CENTRIC INDUSTRIES

by [Franco Gasperoni](#), AdaCore, Inc.

11.1. OVERVIEW

Free markets have come a long way since the medieval marketplaces, and one of the farthest along is the combination of FLOSS (Freely Licensed Open Source Software) and COTS (Commercial Off-The-Shelf software). A free and open market provides a common meeting place where customer and supplier interests can intersect and align, and the combination of FLOSS and COTS is unique in providing the best way to get a virtuous cycle up and running, especially for software designed for the long term, where safety is a critical component.

Let's look closer at an example of this free market at work: AdaCore and GNAT. The keystone to this successful combo was actually laid by the terms of the development contract between the US Air force and New York University: the contract stipulated that the software developed under the 3 million dollar engineering contract would be released under a GPL license, and that copyright of the software would be assigned to the Free Software Foundation. The freedom of the software is the backbone of AdaCore's business model: supplying a leveraged service: top notch support piggybacked on a ready-to-use COTS software package.

AdaCore uses a subscription model: revenues come from recurrent annual subscription fees, based on the level of support and the number of users to be supported, rather than the high initial purchase price model of proprietary closed source software. AdaCore's COTS products aren't available without the support -- that simply isn't AdaCore's business. AdaCore's support is the conduit between the users of their COTS product, GNAT Pro, and the actual developers of that software product.

GNAT Pro is a complex software package, a robust and flexible Ada development environment based on GNU GCC compiler technology. From its core compiler to its debugger, libraries and Integrated Development Environment (IDE) all components are entirely based on FLOSS. The Ada language is widely used in safety-critical applications, and AdaCore offers the GNAT Pro High-Integrity Edition, with runtime support amenable to safety certification.

AdaCore develops and maintains the free software, and packages it with documentation into a polished, well-supported COTS software product. AdaCore's products compete with other solutions from proprietary vendors who also offer polished COTS Ada development environments. But there's one difference: AdaCore also competes with anyone else who wants to repackage and sell the core of AdaCore's COTS product. And that's where the free market comes into play.

Software infrastructure investments have a hidden additional cost -- the cost of switching. In addition to the up-front costs of buying a proprietary software package, the customer is at a complete disadvantage when negotiating with his software supplier. Support costs and service levels provided are not based on the value to the client, but rather on what the supplier can demand, knowing full well what the cost would be to the client if he has to switch his COTS software base. Since the proprietary software copyright holder has a monopoly on the use, copy, and modification of his source code, the customer cannot shop around for a better deal, or higher-level support. Customers become locked in to their software suppliers. The market is flawed, and cannot function as a nexus for free exchange.

FLOSS COTS software changes the game, and levels the playing field. Suppliers have to offer greater value to customers in order to keep them. Niche players can thrive in a software ecosystem where their specialization offers premium value to their specific clients. Development, i.e. program modification, is decoupled from support and certification material. The copyright monopoly lock-in is broken, freeing the user and providing choice.

FLOSS and COTS converge, to provide the ultimate free market. Exchanges from customer to developer provide the building blocks to improve and evolve the software itself. A virtuous circle is established.

The driving force behind COTS is to share costs, to spread the investment and risks of software development over more entities that have similar needs. The AdaCore business model can be extended along another dimension: software co-ops. Software co-operatives group several companies to share resources and know-how, in order to develop and support software over the long term. Working together, they can reduce costs for developing certification material or safety or security, and more generally, improve software quality for the co-op members. Co-ops extend the notion of open source: the software developed in a co-op need not necessarily be open source: a vendor could, for example, put his software under a special license, and make it available to all ATM operations in Europe, to get the co-op started.

This could be directly applicable to ATM software. Let's imagine, for an instant, what would happen if each ATM centre in Europe, when they put a software development bid out, were to stipulate that the licensing rule is that the sources should be available for the ATM centre to use as it pleased. At that point, resources from each centre could be cooperatively pooled to take the best of each development effort, and create the ultimate ATM system.

Today, the difference between OSS and Free Software is philosophical: Open source is about a development methodology, while Free Software is about license freedom. FLOSS copyright terms impact two orthogonal licensing issues: freedom for users, as discussed above, and a free development community. Some companies have tried to curtail developer freedom, like Apple's original APSL version 1 license. But the community will not invest its efforts into projects where their freedom is limited, and to be able to leverage communal development forces, Apple had to back off on the restrictions they could impose, in order to remain competitive in attracting developer community interest.

Four examples of FLOSS freedom:

1. A third party unconnected with AdaCore took GNAT technology to produce a compiler for a radiation-hardened 32 bit chip. AdaCore has ended up competing with our own technology on this market.
2. An independent tool vendor uses AdaCore technology to provide static runtime error detection for Ada. They built in-house expertise and support it themselves.
3. A library with a 32 socket limitation (mainly for historical reasons) was a problem for one customer. They simply made the changes themselves, since they had the source code and the rights to modify it.
4. A specific tool was needed by EUROCONTROL Brussels. To be cost-effective, this tool had to leverage AdaCore's compiler technology. AdaCore submitted a bid, but didn't get the job. It turned out that EUROCONTROL took a comparable bid from another vendor, just because they could: they weren't locked in.

FLOSS is an unqualified plus for the COTS customer, and helps the customer's of negotiating strength. Although the situation would be very different if free licenses had been the norm from the onset, customers should now lobby their vendors to at least release the source code to themselves, the customer. Optimally, the contract terms would require full FLOSS licensing of the development work. AdaCore is a poster child for the win-win opportunities for vendors and customers that FLOSS licensing offers.

A final note on patents: The discussion above has been based on copyright as the fundamental attribute for economic control of software. If US-type software patents are part of the game, the whole system collapses. At that point, a FLOSS implementation of a given software system could be blocked by the patent holder for the duration of the patent. From a practical point of view in the software field, that duration is eternity.

11.2. DISCUSSION

from 26'00" to 35'00" (9'00")

[P. Kappertz](#): *Can I get your software without paying anything?*

[F. Gasperoni](#): *There are two answers: yes you can, but we don't have to. We could shut down the libre site, and simply provide for our customers only, but they could redistribute it.*

The license that we used on our Libre software is made for people who do Free Software development. We call the libre version the GPL version. The license we've used on the Libre version, for a qualified, complete piece of software, is such that it's made for free software development.

It has some legal implications. If you're using it to develop open source components in Ada, then using GNAT Pro is great! Take the example of the trajectory system: you have that software, assuming it was written Ada, what you could do is put out the compiler for all the platforms you wanted to support on a web site, so that developers would not only have the software and the build procedures, but the tools to keep developing the way you did the development.

In order to focus on the free software, there is no reason for us to make it available for people who are not in the free software game. Our license is such that it is that when it is downloaded from the net, it is for free software or open source development.

[H. Lueders](#): *Concerning the financial model, how do you deal with variable levels of support requirements? Customer demand may differ from one moment to another, while the subscription model is a flat fee over the period. The fee also depends on the number of configurations or platforms you want supported, whether it is native (Linux, Solaris, ...) or embedded.*

[F. Gasperoni](#): *Cost is based on the number of developers. The price is not the same for five users as it is for fifty. It goes up or down as your number of developers increases or decreases.*

[H. Lueders](#): *Does this mean that in addition to the retainer fee there is an additional fee for specific services?*

[F. Gasperoni](#): *No, the price is very simple. For example, you say: we support 28 configurations, we have 15 developers, and we want GNAT Pro for Solaris, HP/UX and Linux. Then you have a price based on your request. We have a matrix based on the number of users and configurations. Within this fee, there are an unlimited number of questions which may be serviced.*

[P. Johnson](#): *How do you manage customers, so that they don't try to take advantage of a 'blank check'?*

[F. Gasperoni](#): *We manage it in the following way. We sell a COTS product, so the customers can say, "How do I use it? What is the best way of using it?" There can be problem reports, we fix it and make it available in the next release.*

It's true we have an unbounded number of questions. But there is a great disparity between customers. Some customers ask many questions at the beginning, then they build expertise inside, and they ask fewer questions. It all works out. It's a statistical way of handling it.

To put it in perspective, we're about a ten million dollar company. We have about 55 people working on both continents. So it's both a small organization, and a large organization. Out of the 55 people, about 45 are technical people, engineers, PhDs, people who have developed Ada. Since we're on two continents, we have the same IT needs as a large organization.

J. Feller: *You talked about two types of monopolies created by proprietary licensing, one was a monopoly for creating a derivative work, the other for effectively supporting the product. The examples you gave as why AdaCore is not a monopoly, all are examples of companies creating derivative works. And that, for me, has nothing to do with your business model; it has to do with the license that you chose.*

Without naming anybody, is there anyone competing with you for your core revenue model, that is to say, is there anyone out there receiving monthly subscriptions to answer questions about GNAT Pro?

F. Gasperoni: *On the versions from which they branched off?*

J. Feller: *No, on your version.*

F. Gasperoni: *Not that I'm aware of. It could be, but you see, that's the beauty of the story. There are other examples where they customized versions of our software, but think about why. We keep our prices and our infrastructure to be cost-effective. If we were outrageously expensive, there would be loads of people out there.*

J. Feller: *I'm not so sure there would be.*

F. Gasperoni: *Why do you think so?*

J. Feller: *Because in practice that doesn't happen. If you've been sitting on the credentials of "we're the creators of this technology", and no one has stepped up to say, "well yes, but in the intervening decade I've achieved as much competence as they", then it's not a software licensing issue, it's something else going on there. There's a global community, why aren't there any experts outside of AdaCore?*

F. Gasperoni: *That's a fair statement, but you must realize that it's sort of a chicken-and-egg problem, because if we raised our prices outrageously, I can tell you we'd be out of business quickly, in a variety of ways, of which that one is a possibility.*

There is a case in the community: wouldn't it be nice if AdaCore provided support for a single developer for 500 euro, but we know that isn't going to fly. And we said, "Fine, if somebody else wants to take it up, fine, we don't think that business will survive for a long time.

The fact of the matter is that we haven't seen much competition because it's tough to serve a very specific market, and a very technical market, at those types of prices.

11.3. SALIENT POINTS

- AdaCore Inc. sells COTS based on FLOSS.
- This company is born from a single/simple OSS decision from the US Airforce.
- It uses a subscription model always including a support without initial peek cost.
- AdaCore uses Open Source for safety critical applications.
- Other COTS providers can repackage AdaCore FLOSS-based COTS products.
- FLOSS plus COTS means no one entity has a monopoly for changes, support, and certification material.
- For the customer, the absence of monopoly aligns customer and vendor interests, to provide high-quality products and services at competitive process.
- While an Open Source development methodology has its plusses and minuses, the license freedom FLOSS guarantees for the customer is an unqualified plus.
- Customers should lobby their vendors for at least release of the code they pay for to themselves as customers, or preferably public release of the code under a FLOSS license.

12. OSS AND IPR EVOLUTION IN EUROCONTROL

by [Burkhard von Erlach](#), EUROCONTROL

12.1. OVERVIEW

Free software is certainly not new, but awareness of it fairly new at EUROCONTROL. This presentation covers the current policy of EUROCONTROL and highlights why EUROCONTROL as an international organisation needs to be cautious and is not as free as industry. We'll list a few conditions under which OSS might become more successful in ATM.

First off, let's define the terms we'll be using.

Stallman's work in the 80s defined Free Software, with free meaning "libre", guaranteeing four freedoms: freedom to run the program, freedom to study and adapt the program, freedom to redistribute copies of the program, and freedom to improve the program and distribute the improvements.

In the late 90s E. S. Raymond defined another term, OSS, and articulated why he believed that open source licenses resulted in higher quality, less expensive software.

To be recognized as such, OSS needs to fulfil the following criteria:

1. Free Redistribution: the software can be freely given away or sold.
2. Source Code: the source code must either be included or freely obtainable.
3. Derived Works: redistribution of modifications must be allowed.
4. Integrity of the Author's Source Code: licenses may require that modifications are redistributed only as patches.
5. No Discrimination against Persons or Groups: no-one can be locked out.
6. No Discrimination against Fields of Endeavour: commercial users cannot be excluded.
7. Distribution of License: The rights attached to the program must apply to all to whom the program is redistributed without the need for execution of an additional license by those parties.
8. License must not be specific to a Product: the program cannot be licensed only as part of a larger distribution.
9. License Must Not Restrict Other Software: the license cannot insist that any other software it is distributed with must also be open source.
10. License Must Be Technology-Neutral: no click-wrap licenses or other medium-specific ways of accepting the license must be required.

The definition of OSS is more detailed and requires more conditions to be fulfilled than free software. One can therefore say that all OSS is free software, while the inverse is not necessarily true.

For our purposes today, we'll define Open Source as software that is built, developed, and enhanced through public collaboration, and which is free in terms of unrestricted access to the source code. This does not mean that it is necessarily free of charge, nor does it mean that it is free of copyright.

Speaking of copyright, copyright is the right created by an author of publications in the literary, scientific and artistic domain, whatever may be the mode or form of its expression (including software). Under continental European law, copyright is automatically created and belongs to the writer/developer of software and is inalienable.

Copyleft uses copyright law in order to ensure that every person who receives a copy or derived version of a work, can use, modify, and also redistribute both the work, and derived versions. This achieved mainly through the General Public Licenses, which aims at ensuring that users adhere to the 'conditions' listed earlier and to make sure that further developments are made available as open source as well. It should be noted, however, that private copies and developments made for private use and which are not further distributed do not fall under the obligation of redistribution.

Patents are probably the biggest enemy of any OSS movement, as they tend to block the free use of inventions by third parties. The approach to patentability of software is a rather different on both sides of the Atlantic. The United States have taken a more 'liberal' approach and are granting regularly patents for software as such. Software patents in Europe have suffered a setback, in early 2005, by the Parliament's rejection of a new directive. The European Patent Convention explicitly excludes the patentability of computer programmes (software) as such. Software is, however, considered to be patentable if it has a technical character.

Finally, we should mention the public domain and freeware. Public domain is any material that is uncopyrighted, i.e. whose copyright has expired, or is uncopyrightable. This can include government publications (depends on the state), factual data (measurements, heights of mountains, telephone numbers etc.), jokes, titles - and ideas and which are available at large. Freeware is software made available free of charge. It is copyrighted by its developer, who retains the rights to control its distribution, to modify it and to sell it in the future. It is typically distributed without its source code, which prevents modification by its users it therefore is certainly not open source.

Now we can take a look at the current situation at EUROCONTROL.

EUROCONTROL as the European Organisation for the Safety of Air Navigation, an international intergovernmental organisation is constrained by a range of rules and regulations most of which are contained in the EUROCONTROL Convention and the Contract and Finance Regulations. In 1995 intellectual property rights guidelines were established and adopted by the governing bodies of the EUROCONTROL Organisation.

The guidelines can be summarized as follows:

1. Software created, developed and produced by or on behalf of EUROCONTROL (under a contract) become and remain the exclusive property of EUROCONTROL, who is free to seek appropriate legal protection of the intellectual property rights in that software.
2. Where software is created, developed and produced in the course of a co-operation programmes resulting intellectual property rights would be owned by all parties involved.
3. Software nowadays regularly builds on pre-existing software (background software) developed by contractors prior to 'contracts'. Such background software is reused in the production of a deliverable. Of course ownership for such pre-existing software cannot be claimed. But EUROCONTROL has to try to obtain all relevant rights to use and distribute this background software together with the foreground software specifically developed for EUROCONTROL.

In terms of licenses, EUROCONTROL makes available to its Member States and the appropriate ATC entities software, subject to the conclusion of appropriate license agreements usually free of charge. Restrictive licenses are granted to third parties for software, subject to appropriate, the license agreements shall contain a clause ensuring a return of information (feedback) to be used by the Organisation for its own purposes and subject to the payment of license fees/ royalty fees.

[R. Schreiner](#): *Isn't it in complete contradiction with what we are discussing today?*

[B. von Erlach](#): *Very much so, but it's the policy today. We are in an early stage of open source and we are looking in it.*

The Intellectual Property Advisory Group composed by a number representatives of services of the Organisation monitors the application of the Guidelines and discusses cases on IPR requests which lie outside the policy and issues its opinion the Director General.

Since EUROCONTROL is a not profit oriented, selling or commercialising software is therefore not its prime objective. Bearing in mind our special status, EUROCONTROL needs to be a cautious, since it operates under a few specific constraints. EUROCONTROL cannot and does not wish to compete with private industry, since this could and would be seen as unfair competition. In addition, software is very rarely developed from scratch and is often based on pre-existing proprietary software. It is not possible to make such a tool (often involving third party software) available as OSS without facing severe risks of violating third party rights.

While commercializing software and thus patenting are not EUROCONTROL's prime aim, it is important to ensure that third parties are not patenting software or developments based on our ideas, works and would thus bar the ATM community from using such developments (or allow it only at high prices). Hence, we should pursue a large openness, as this would "destroy" novelty, which is one of the indispensable criteria for patenting.

[H. Lueders](#): *This means you don't own any patent on software by yourselves.*

[B. von Erlach](#): *No, we don't.*

Let's now look toward the future. Other areas seem to have adhered more quickly to the OSS movement than the ATM environment. Several factors may explain why. ATM is a highly safety critical environment and reluctant to use software which has been developed/modified by third parties. Open source development can be perhaps too quick in an ATM environment which needs a certain stability. Warranty and liability issues are unclear. Current arrangements would no longer be applicable if open source was to be used and this could have an effect on liability/insurance.

[P. Johnson](#): *Microsoft sends you an agreement saying that the software is warranted for a period of ninety days.*

[J. Seifarth](#): *"No guarantee for merchantability and/or fitness for any particular use."*

[B. von Erlach](#): *They do write it down, but I am not sure that in a real Court case, it could be applicable.*

[J. Seifarth](#): *That's why it says underneath: "Depending on the State, this warranty may not be applicable."*

[J. Feller](#): *Core to the point, Microsoft Word does not keep airplane separate from each other.*

[J. Seifarth](#): *No but a SQL database...*

[J. Feller](#): *That's true!*

Maybe the open source is not as successful as foreseen, because the ATM field requires special skills and expertise which is rather expensive to acquire and may perhaps not be found in large numbers in the open source community.

In addition one has always to bear in mind contractual and/or legal restrictions that hinder a free distribution of software with open sources. There are often pre-existing software/rights which EUROCONTROL is contractually committed to respect, they cannot be put as open source, nor can any development be based on them. There are also legal restrictions, the best known are probably the US Export Restrictions.

P. Johnson: *As soon as it published, the restrictions do not apply.*

O. Robert: *They do try to restrict it to certain countries like Iraq...*

B. von Erlach: *Yes, Iraq, North-Korea, Sudan, Iran, ...*

M. Den BESTEN: *What about the French restrictions on cryptography?*

O. Robert: *They were relaxed a few years ago, the consensus being: "Don't tell anybody, and nobody will bother you!"*

M. Den BESTEN: *It sounds like a very Dutch compromise!*

B. von Erlach: *The US restrictions are the most known, but they are others.*

Nevertheless, OSS has obviously also advantages, which could prove interesting to ATM. It allows a quick efficient change, especially for bug fixing. Since there are usually no license fees, it is cheap.

There is no vendor lock-in. Finally, the entire issue of background and foreground software would probably become obsolete.

P. Kappertz: *In what sense do you think open software is cheap?*

B. von Erlach: *Well certainly in acquisition: you do not have to buy license fees.*

P. Kappertz: *In Germany, lots of communities have evaluated costs basing their software on Linux or Windows, and the differences over the lifetime are usually negligible. It's a very short term view if you only look at the acquisition.*

B. von Erlach: *You probably have to look the all life cycle, and make an analysis of the full business case before embarking in open source.*

P. Johnson: *The reason we're working on it in NATS is because there is no lock-in. We call it "Destiny Ownership".*

J. Seifarth: *Destiny Ownership!*

How could open source work for research and development? In principle nothing would really change from a procurement/contract point of view, as we are entitled to put it into 'open source'. It would be, however, prudent and fair to indicate in our contracts to make suppliers aware if we were to adhere to open source movement. There is a certain risk of scaring industry that their intellectual property assets would become available to the public at large. This would probably impact the price and the way software is being developed and made available. Some companies are extremely worried to see their know-how disclosed to competitors.

In any event to be successful and recognised in OSS the following conditions would need to be met:

1. There needs to be an active, motivated and capably community interested to pursue this.
2. There needs to be a well known easily accessible forum / portal.
3. The community would need to be guided and administered. This would require an authority which is, and is seen to be, neutral, competent and reactive/proactive.
4. For the ATM community to enhance credibility and acceptability, any initiative would need the active involvement of ATC entities of the Member States, and possibly industry as well.

As an international organisation independent from industry and/or particular Member States and thus any particular interests other than the safety of air navigation, EUROCONTROL might have a role in such an OSS initiative in ATM.

In conclusion, open source as a concept is quite interesting, and certainly worth further study. However, there still will be no to avoid having to study the implications so as to ensure that no third party rights are violated. EUROCONTROL is starting to direct its attention the open source initiative and it is not by accident that this round table takes place at the EUROCONTROL Experimental Centre organized by our Innovative Research Unit. While policies take a bit of time to be accepted and changed, there is hope that this may happen in the future. The nature of R&D and Experimental Centres would probably be the areas where an open source initiative seems the most promising as a start.

12.2. DISCUSSION

from 21'30" to 30'17" (8'47")

M. Bourgois: *What holds us from putting everything we have done in the past on to open source? Is it a question of fairness to the suppliers?*

B. von Erlach: *We don't own the entire tool, only parts of it. Theoretically, we could probably do it, but it depends on the pre-existing software.*

A. Engelfriet: *In the past, when IBM developed software for a customer, the customer like EUROCONTROL insisted in having the complete control. It was a problem for IBM because they could not sell it to anyone else. IBM is now basing their solutions very heavily on open source, so now they can not tell their customers: "Look, we'd love to give you the copyrights, but most of it is GPL". That's something to take into consideration. If you go more towards open source, your suppliers will base themselves on open source.*

M. Bourgois: *That would be no concern. We want to avoid that other close it. We do not want to close it. So that is not a threat to us.*

B. von Erlach: *You have to bear in mind that software is considered as an asset here. It is in the books with a certain amount of value. Obviously, all the regulations we have aim at maintaining these assets at the moment.*

C. Garcia Avello: *What about liability? If you use OSS, are you still liable?*

B. von Erlach: *You try not to be. It might be difficult to trace back who has done something wrong. You can not deliberately put ill-functioning material on a site and think you can walk away without liability. However, all the OSS available always exclude liability since it's in an experimental stage; but if it's taken over and maintained by a company, this company should take some responsibility. Up to now, there has never been liability a lawsuit involving OSS, but it might happen in the future.*

P. Johnson: *I have never heard about an OSS maintainer being sued for liability.*

H. Lueders: *You said that you do not need patents since you are protected as soon as the software is published and the idea is no longer new. But how do you protect your software from being stolen? Do you think that a copyright is a sufficient protection? The technical contribution is the most expensive investment and it's the only element you can not protect with copyright. Wouldn't patent make a lot of sense here? We should not only consider the steeling of the US and talk about China and Japan. Japan has just opened its first patent court on 1st April and based their whole innovation policy on IPR.*

B. von Erlach: *In general, we are mostly involved in software development. The software we develop at the moment could not be subject to a patent under European law anyway. We are keen to make sure it is at least published, so others can use it as well. This lead to a de facto harmonization.*

F. Gasperoni: *If people steal it, then they break the law, the same way they would break the law with a patent, so copyright is sufficient.*

A. Engelfriet: *I think a patent is only useful to protect against someone that wants to copy your functions.*

M. Bourgois: *If people steal the software, why not? If they use it, we achieve our goal as European organization.*

B. von Erlach: *Most of all, once we have paid for a software development, we want to avoid having others selling that software again to us.*

12.3. SALIENT POINTS

- The definition of OSS is more detailed and requires more conditions to be fulfilled than free software. One can therefore say that all OSS is free software, while the inverse is not necessarily true.
- Under continental European law, copyright is automatically created and belongs to the writer/developer of software and is inalienable.
- Copyleft uses copyright law in order to ensure that every person who receives a copy or derived version of a work, can use, modify, and also redistribute both the work, and derived versions.
- Patents are probably the biggest enemy of any OSS movement, as they tend to block the free use of inventions by third parties.
- Since EUROCONTROL is a not profit oriented, selling or commercialising software is therefore not its prime objective.
- EUROCONTROL cannot and does not wish to compete with private industry, since this could and would be seen as unfair competition.
- It is important to ensure that third parties are not patenting software or developments based on our ideas, works and would thus bar the ATM community from using such developments.
- Open source development can be perhaps too quick in an ATM environment which needs a certain stability.
- It would be, however, prudent and fair to indicate in our contracts to make suppliers aware if we were to adhere to open source movement.
- The nature of R&D and Experimental Centres would probably be the areas where an open source initiative seems the most promising as a start.

13. PERMISSIVE VS. RESTRICTIVE OSS LICENSES

by [Delphine Prieur](#), INRIA

13.1. OVERVIEW

Legal aspects of Intellectual Property protection vary from country to country. This presentation discusses OSS licenses, often from a French point of view, and deliberately avoids the issue of software patents, instead focusing on copyright protection. From the onset, two points are worth noting: a proprietary license can be more permissive than an OSS license, and OSS licenses are an original way to use copyright.

Software is an IP tool protected by copyright. The legal protection of software is divided between moral rights (*droits patrimoniaux*) and economic rights (*droits commerciaux*). Copyright law grants a monopoly on the work to the author for these two types of rights.

Moral rights in software is essentially to be able to oppose modifications when they would be harmful to the honour of the author, or his reputation. Economic rights are usage rights such as reproducing, translating, modifying the software. The duration of the monopoly is very long: almost 70 years after the first display. Protection is automatic; the author needs no formalities to be protected.

So then, what exactly is a license? It is a charter of rights and duties, a contract in which an author, while preserving the whole property of his work, confers to a certain extent, the possibility to a licensee to implement his rights. Some people consider that a license is more or less permissive when the possibilities offered are higher, or vice-versa. For OSS, another way to express this dichotomy in a more positive way: the rights granted to the licensee have to be implemented with more or less reciprocity. In other words, the more freedom you have, the more responsibility you must feel.

Two categories of licenses are frequently identified: proprietary licenses and Open Source licenses. Without delving into the differences between the Open Source Initiative (OSI) definition and that of the Free Software Foundation (FSF), we can note that OSS licenses are based on copyright law, but used in an original manner to produce the desired effects. Often called copyleft, OSS uses copyright. It is different from the public domain. Whether it's because it's no longer protected, or because the author has chosen to place his work in the public domain, the point is that no one can reserve property rights on the public domain. OSS copyleft, in contrast, exists solely because of copyright law.

The natural legal risk for Intellectual Property Rights is counterfeiting, a notion that is left to the appreciation of the judge, on the basis of many different criteria that vary greatly from one country to another. Proof of counterfeiting is usually based on resemblances rather than on differences: copied code (the easiest case), sequences of screens, order and logical organization of functions and control structures, in one case even spelling mistakes in code comments. Punishment for counterfeiting can range from monetary compensation to prison time.

In the world of OSS licensing, despite there being around 50 different licenses out there, we can concentrate on three major ones right now: 1) GNU/GPL, the most visible, widely used and also most criticized, 2) CECILL, a recent GPL-based license produced by CEA, CNRS and INRIA to fully comply with French law, and 3) the new BSD license, permissive and succinct, as opposed to the GNU/GPL. The current trend is to try to reduce this number, and this is a hot topic within the community.

Let's look at these three approaches to OSS licensing a bit more closely:

- The GPL guarantees that you can use, modify, and distribute the code. In compensation, copyright notices must be kept, and users can obtain the source code, however, the GPL offers no warranties, and no responsibilities. The GPL is often called "viral".
- The CeCILL is French modification to the GPL, a recent license developed by CEA, CNRS, and INRA, and offers a different approach to warranties and liability, in order to conform to French law. For example, under French law, you have liabilities and cannot exclude some warranties once the product is aimed at consumers. The approach used in the CeCILL is to position the license as a contract between professionals. This license is currently in the process of OSI approval.
- Finally, the current BSD license allows use, distribution and modification with or without redistribution of the modified source code.

O. Robert: *The older BSD licenses had a so-called "advertising clause" which stipulated that code using the BSD licensed sources had to have a screen advertising that the code contained parts licensed under the BSD. Current BSD versions no longer use this clause. Compensation resides in keeping copyright notices and license terms intact. The BSD license is the canonical example of a non-copyleft OSS license.*

A few points should now be noted: first, the legal framework has not yet been tested. Few cases have been tried, and there is a movement from considering them as natural risks (counterfeiting) toward examining them for contract compliance. Those few cases tried have not attempted to validate the GPL, rather they have targeted whether the users have been complying with the license.

Lloyds insurance is studying insurance coverage of up to \$10 million for damages, including profit losses related to non-compliance with an OSS license. In some cases, the policy could cover the cost of repairing the code found to infringe on an OSS license.

With regards to the number of IPRs, conflict brought to court are rare, lots of them are solved by arbitration or transaction and are kept secret. No one has an interest in invalidating an Open Source License, the consequences would be worse than the initial problem. Besides that, these legal issues are complicated by the multiple nationalities in presence, and the same is true for liability.

A crucial point is that the multitude of licenses generates compatibility problems between these licenses. For example, BSD is compatible with GPL, but GPL is not compatible with any other license.

O. Robert: *GPL is not even compatible with itself.*

For OSS-based business models to be safe, new concepts were developed like OCC management, tests of source code infringement. A new prototype is being developed by the INRIA, to diagnose the legal status of a source code.

One of the extraordinary forces of OSS is the community. The OSS community can often react to programming bugs within 24 hours. Now the OSS community is defending the rights of the authors with tools like the web site gnu-violations.org, based on the compliance to the GPL license.

In conclusion, when choosing a license, it not important to consider whether it is permissive or not, but to ask the relevant questions such as: what are your intentions for development, distribution, and modification of your code? What is the status of the contributions of people? Who will use the code, why, and to do what? It is important to keep in mind that these intentions may change, so you should use a flexible model.

13.2. DISCUSSION

from 15'00" to 15'29" (29")

[J-D. Frayssinoux](#): *About your slides, they will be available?*

[D. Prieur](#): *I was late, but I gave them to the organizer.*

13.3. SALIENT POINTS

- Software IP protection is based on copyright, used in an original manner to produce the desired effects.
- OSS licenses can be differentiated on the basis of required levels of reciprocity in contributing code changes back.
- Although the natural legal risk for copyright protected IP would be counterfeiting, current legal action has been based more on contract compliance.
- The multiplicity of OSS licenses can cause conflicts, and special attention must be paid to the result of combining software packages licensed under different OSS terms.

14. WORKING WITH OSS LICENSES IN PHILIPS

by [Arnoud Engelfriet](#), PHILIPS

14.1. OVERVIEW

The IP department of PHILIPS has been working with OSS for the past 4 years. We'll discuss how PHILIPS use open source in products, mainly in consumer electronics, and what we did to get the rules right, and to make sure everyone is following them. We'll also discuss how to make effective choices concerning open source -- what to open source, when and when not to open source your software.

Let's first review the main open source licenses, and avoid loaded vocabulary like 'viral'.

- **free-for-all licenses**, like the BSD, are simple: just give credit for the original software, and use it however you wish,
- **keep-open licenses**, like the LGPL and the Mozilla license, allow you to combine them with other code under other licenses, but require that you give back any modifications you may make to the open source code,
- **share-alike licenses**, like the GPL, that require you to contribute not only the code to modifications, but the code to extensions as well.

[F. Gasperoni](#): *You must only release your code if you distribute products based on the modified code. If you do not distribute the modified code, but only use it internally, you do not have to release your modifications.*

[A. Engelfriet](#): *PHILIPS, of course, sells its products to millions of people, and does fall within the distribution requirements.*

[F. Gasperoni](#): *EUROCONTROL uses ATM systems that are not sold to the general public typically.*

[A. Engelfriet](#): *Indeed, if you are going to use it internally for your organization, that's fine: you do not have to tell anybody, you do not have to share anything. However, I am approaching this from the point of view of a distributor.*

[P. Johnson](#): *Even PHILIPS can use and modify Linux internally.*

[R. Schreiner](#): *What is "internal" in this sense?*

[A. Engelfriet](#): *That's a very good question, especially for a multinational company.*

[F. Gasperoni](#): *The reason why people do not like the word viral, is that if you PHILIPS decide to take Microsoft Windows and put it into your product, viral or not viral, you couldn't do that, full stop.*

Therefore, while the technical people in the company do not have to know the details of the different licenses, they do have to know the potential impact of choosing to use software with a share-alike license.

PHILIPS use open source in products, mainly in consumer electronics, like hard disk DVD recorders, universal remote controls, and web tablets. Our new line of wirelessly connected devices, Connected Planet, has many devices on embedded Linux running 802.11 wireless communications for audio and video streaming. PHILIPS also develops chips like the Nexperia line, with their compiler, tool chain, and Linux ports all available.

The main reasons for Open Source usage in PHILIPS is cost reduction and faster time to market. It's very seductive: developed, useful code for free. We're looking at making our use of open source more systematic, and are examining the possibility of releasing some of our own code as open source. Before committing to using open source, we decided to first establish an open source policy.

An open source policy is necessary because suppliers and customers are increasingly using OSS, and OSS can have an impact on PHILIPS' intellectual property rights. For example, PHILIPS holds many patents, which they do not want to license for free. Using open source can imply certain patent obligations, which will have an impact on the patent holder. We want to be sure we know what we're doing, and when we're doing it. It shouldn't be a decision by the individual engineer; it must be taken at the product design level.

F. Gasperoni: *Arnoud, is it considered distribution if you sell a VCR with embedded OSS?*

A. Engelfriet: *It would most definitively.*

F. Gasperoni: *Because one question that was asked is: what if I put some GPL software on a missile. The question that was asked to Stallman is "Is this considered distribution?" The answer was: "Of course, not".*

A. Engelfriet: *If you are a missile dealer, and you sold it, then it is a distribution.*

P. Johnson: *If a missile manufacturer sells to the DoD, that is a distribution.*

A. Engelfriet: *The DOD has the right to demand source code.*

O. Robert: *You could put the source code into the missile ;-)*

F. Gasperoni: *It's the same for the source code inside the VCR: it is in the ROM.*

A. Engelfriet: *Of course, it must be accessible to the outside world. We have a product with a hard disk, and we put the source code of Linux on the hard disk. If you read very carefully to the documentation (but nobody does that) you will see that this product uses Linux and it is there on the hard disk. For other products, we have a CD-ROM for documentation where the source code is included.*

This is an interesting support issue: clients may wonder why there's 20 MB of disk space used, when they want to record on that space. Few people are going to bother and that's a practical frustration for the product managers: they say that nobody wants the source code. The important thing is that the source code is there, as required by open source license compliance, available to anyone who wants to use, study, modify and improve it.

P. Johnson: *Could it be on a web site?*

A. Engelfriet: *You cannot actually. The license says either you ship it or you make a written offer to give it to anybody who writes to you. The source code has to accompany the product. GPL version 2, article b), section 3) states that if we put a binary on the product, the source code must accompany the product.*

Anyway, PHILIPS' policy on open source allows unrestricted use of OSS internally. What tends to happen is that an internal research prototype becomes a product. So be careful with that.

PHILIPS is a company that manufactures and commercialize products; the use of OSS in products is fine as long as it does not involve differentiating technology, i.e. an added value feature, and if it does not open valuable patents. We use open source only where it makes sense for us.

If we release something in open source, it has to benefit both PHILIPS and the community; There is no point in starting a competitor to an existing open source project.

Finally, every decision about OSS must be taken jointly by the IP&S and the Open Source Advisory Board.

Engineers will put open source in products, suppliers will ship open source, even without you knowing it, and end users will combine existing code with open source. Your own developers may participate on outside open source projects; you must be prepared to deal with that. It's thus vital for the company to have a clear vision of the implications the use of Open Source might have in terms of legal aspects.

You can benefit from open source. There are two general ways: consumption or participation.

You can decide to buy some software from a third party, you can write some yourself, and use open source for others. You can even decide to buy open source from a third party, for the support they provide. This is passive use of open source.

More proactive use might involve participating in an open source project, for example to make it work on your embedded product. You can then contribute those patches for common maintenance, and adopt this existing project. You can also start your own open source projects, and build community around them. This can be a lot more powerful, since you can share costs and contributions, however it requires more discipline: you must be prepared to expose your code for the world to see.

Once you have a policy, you will need a review board to examine and decide on each individual case. You need a place where people can come and ask what is allowed and what is not. Set up a group which can represent all your business units or product divisions, and preferably also involve the legal and IP departments, to consolidate your expertise in this area. You may want to set up set up liaisons with purchasing, customer services, PR communications and HR. For instance, if we hire an open source engineer, he could be in trouble if the contract says that everything he writes is owned by PHILIPS. Also, the customer service flowcharts must include answers about the source code provided on the CD-ROMs.

The OSS policy should actively address both customers and suppliers. By default, the contracts clauses should be "Don't give us any OSS and don't make our software OSS". This must be clearly stated and standard contract clauses should be created to address this issue. While this may seem draconian, it really is necessary to get people thinking about what they're really doing. If there must be an exception, it's clear, transparent, and documented from the onset.

Open source usage by subcontractors can have unexpected implications, even with free-for-all software: learning too late that some subsystem incorporates BSD licensed code could require reprinting all the documentation to add the credit notices. You must determine how you will distribute the source code, with a web site, or keep a stack of CDs in customer service to send to those that request it, or by distributing the source with the device.

When you use open source in products, you must first determine where you should use it. We can classify product features into 3 different groups, and determine what level of open source is appropriate.

- Differentiators provide added value to your product by unique features. This type of software should be kept closed, for maximum market advantage.
- Baseline features provide the necessary functionality to meet market demand. Use a case-by-case approach to decide whether to make, buy, or use open source.
- Commodity features are essentially invisible to the user. Using open source saves time, effort, and money.

We can now walk through an example of open sourcing decisions in a product that PHILIPS developed: the Active Block IO Scheduling System (ABISS) scheduler. The underlying idea is to have a disk scheduler which you can ask to give you a guaranteed disk access bandwidth for a real-time video or audio stream. The scheduler either returns a guaranteed bandwidth stream, or it tells the caller that the requested disk bandwidth is not available. This significantly enhances the design of a hard disk recorder or video player.

This project was the first in which we deliberately decided exactly which license we would for each component. The system rides on the GPL Linux kernel. Kernel modules which link into it must be GPL as well. The actual kernel module which must be GPL can be kept simple, however, if the functionality of the system is broken into smaller modules. Then each sub-module can be licensed corresponding to its function value as differentiator, baseline, or commodity feature. Thus some modules, like the scheduler and special optimized policies, were kept as proprietary, unreleased code. The kernel module itself was released under the GPL, and the application interface and the userspace policy daemon were released under the LGPL, requiring only redistribution of modifications to the original code.

In this case, the technical architecture was deliberately designed with licensing considerations in mind, and as a result does not require the release of valuable proprietary IPR which implements differentiating software features. Technical choices have to be made with interaction between designers and legal people to ensure that the architecture does not influence your legal options. This requires people who understand both technical issues and law.

14.2. DISCUSSION

from 28'00" to 32'00" (4'00")

M. Michlmayr: *For your DVR, you obviously had to make changes to Linux, and because of GPL you had to make these changes available. Did you actively try to integrate these changes with the Linux kernel? Do you have anything in your policy to address this issue?*

A. Engelfriet: *Actually, our policy actually says that if the project would benefit from the changes, we should give these changes. If they are specific to one of our products, that doesn't make sense.*

M. Michlmayr: *Do you take the long-term effects into consideration? If the changes are not integrated with the Linux kernel, in a few years time, the work will have to be done all over again.*

A. Engelfriet: *There is a conflict of interest there: the product must be out of the door as soon as possible but, on the other hand, there is the opportunity of doing something that represents a long-term benefit for the company and the community in general.*

M. Michlmayr: *How do you cope with that?*

A. Engelfriet: *With great difficulty. We try to work together with other companies in the Consumer Electronics (CE) sector. We have several common projects, among which the CE Linux Forum. This project aims at improving Linux in the context of the CE industry, to achieve faster booting times for example. We share costs and the developments on a common kernel.*

F. Gasperoni: *How does this cooperation work?*

[A. Engelfriet](#): *Pretty well. However, it is not so easy to move from a proprietary model to an open-source model. For some, it's quite difficult to share resources with competitors. There is also a lack support from the open-source community in general, but that's because we are in a very specialized niche. There's a vast cultural gap between the world of consumer electronics and the world of PCs. PC guys are often discouraged by the constraints of the CE industry. CE versus PC, it's a difficult match.*

14.3. SALIENT POINTS

- Open source licenses can be usefully classified into free-for-all, keep-open, and share-alike licenses.
- Share-alike licenses require distribution of the modified source code along with the binary image in the device's firmware, or at least making the source code available on written demand. PHILIPS deliberately complies fully with license requirements and ships the full GPL and LGPL code, either on the device hard disk, or as an accompanying CD-ROM.
- The main reasons for Open Source usage in PHILIPS is cost reduction and faster time to market.
- An open source policy is necessary because suppliers and customers are increasingly using OSS, and OSS can have an impact on PHILIPS' intellectual property rights.
- PHILIPS' policy on open source allows unrestricted use of OSS internally, but the use of OSS in products is a completely different issue.
- OSS should be used only if does not involve differentiating technology, i.e. an added value feature, and if it does not open valuable IPR.
- Every decision about OSS must be taken jointly by the IP&S and the Open Source Advisory Board.
- You can benefit from open source by using it. More proactive use might involve participating in an open source project.
- By default, the contracts clauses should be "Don't give us any OSS and don't make our software OSS".
- The technical architecture can be deliberately designed with licensing considerations in mind, and as a result will not require the release of valuable proprietary IPR which implements differentiating software features.

15. TCAS BEING OPEN SOURCE BEFORE THE TERM WAS COINED

by [Garfield Dean](#), EUROCONTROL

15.1. OVERVIEW

Open Source is based on public review of source code. But before code can be written to meet standards, the relevant standards must first exist. To be useful, a Traffic Alert and Collision Avoidance System (TCAS) must interoperate with other units.

[H. Lueders](#): *A glossary would be useful.*

[G. Dean](#): *Most of the things will be explained, even if one of the slides is in Japanese.*

This presentation discusses the similarities between the open source process, and a technical standardization process.

TCAS is equipment on board aircraft that tells the pilot to climb or descend to avoid collision with other aircraft.

[F. Gasperoni](#): *Is it the one that was overridden by a controller in Switzerland and caused a crash?*

[G. Dean](#): *Absolutely!*

[J-D. Frayssinoux](#): *The TCAS was right!*

[F. Gasperoni](#): *Yes, the human was wrong!*

[G. Dean](#): *It depends where you limit the system. Do you include the controller and pilot in the system or not? The smaller technical system was right. The larger social system was not correct.*

The TCAS uses a type of radar, called a transponder, which measures distance with other aircraft and gets altitude information. Based solely on that information, it evaluates collision detection algorithms each second to determine if the aircraft is in danger of colliding with another. The algorithms used are completely defined by pseudo-code and completely identical state charts. Correspondence between the state charts and the pseudo-code has been explicitly and carefully verified.

These pseudo-code and state charts have some similarities with OSS.

The TCAS concept dates back to the 1950s, but it wasn't until the '70s that a real impetus to the program existed, in the form of a US Congressional mandate to build such a system, following a high-profile aviation disaster. Since TCAS systems were to be placed on all aircraft coming into US airspace, the scope of the project went worldwide.

Therefore, the FAA had to develop international standards from the onset. This required cooperation from people all over the world, and they had to get a group of manufacturers, ideally from all over the world as well, to implement the set of specifications which had been defined.

These specifications, these standards, are vital to the system. If one system says "Climb!" and the other aircraft's system says "Climb!" as well, the collision will not be avoided. Coordination and agreed sets of rules must be established. A high-level set of performance standards was created by the International Civil Aviation Organization (ICAO), basically defining the percentage of collisions the system will be able to resolve, acceptable false alerts rates, and interoperability considerations. Displays were not defined, nor many practical details of implementation.

ICAO is a large organization, and the design required committee approval needing consensus, with delegates from organizations in many Member States. In practice, a handful of States were really interested, and a couple of organisations like the pilots' and controllers' unions and EUROCONTROL. Progress was excruciatingly slow, and the committee was not the driver of the process. Momentum for the standards came from the development work in the field, but the committee provided a forum for discussion. It was a check and balance, using input from underlying validation work.

RTCA (Radio and Technical Committee of Aeronautics) did the detail work, including the display descriptions and architectural design. There are a lot of explanatory texts, but the heart of the specification is a hierarchy of detailed state charts and pseudo-code, which manufacturers use to write their actual code in C or Ada.

[F. Gasperoni](#): *Is it the one that was overridden by a controller in Switzerland and caused a crash?*

[G. Dean](#): *Absolutely! A detailed series of tests are used to determine compliance by the manufacturer's code to the specification.*

In the RTCA, the detailed design was committee-based again needing consensus. The RTCA is quite political, very strongly influenced by the FAA. The chairman is under a lot of pressure from different areas: airlines, pilots, controllers, and manufacturers. In theory, anyone is welcome on the committee, but they must show their competence. There are many different viewpoints: some participants didn't even want the system to exist.

Initially, the system was fluid, if changes were good, they'd rapidly be incorporated into the spec. Inputs came from many sources. The FAA fielded teams to develop alternate representations (the state charts), European teams validated algorithms with European data, and many teams were involved in different areas. The RTCA decision-making process wasn't very transparent, and some participants had more clout than others.

Once the system was implemented, however, incorporating changes became much more difficult. Making certification changes and sending technicians to the field costs quite a bit. A single software change, just downloaded to the devices' firmware, would cost an airline around \$ 5,000 per aircraft.

[F. Gasperoni](#): *That's actually quite cheap.*

[G. Dean](#): *If you consider that they are 10,000 aircraft in and around Europe that requires this software, the cost of a single small software change can reach fifty million dollars.*

How does this development compare to OSS? First of all, pseudo-code is not really code. There are at least 5 different implementations of the same pseudo-code. A corollary of this is that stringent testing standards are required to assure proper function and interoperability.

Another difference with many OSS projects was that the individuals participating in this process were getting paid. The various organizations sending delegates had to cover these costs, of course, but the individuals involved received monetary compensation for their efforts. The process was fairly open. Those who made the effort to be heard could be heard.

Although TCAS is a safety system, it is not a safety-critical system. The question remains, would an OSS version of a safety-critical system be acceptable?

[F. Gasperoni](#): *Is this level "E"?*

[G. Dean](#): *No it isn't. It's classified as being level "B", but it's arguable that it's only need to be level "C". If your TCAS system continually generates false alerts, then it's a hazard. But if it falls over crashes and does nothing, you are just back to the safety level where you were right before; it hasn't induced any new risk. If you coupled it directly to auto-pilot, it might be something else...*

Responsibility for the TCAS units lies with the manufacturers. However, the FAA absolves manufacturers of responsibility for design logic flaws; moreover the FAA cannot be sued. The recent accident in Überlingen may actually test this in court.

Suppose there really was a flaw in the TCAS logic (and we know that, in a very small percentage of cases, TCAS will create a collision that was not there previously, in the same way that a very small percentage of people inoculated against a disease will have an adverse reaction and die as a result of the inoculation), who will take the responsibility? The same is true for any form of safety critical software.

[P. Kappertz](#): *Was the fault in the algorithm known before?*

[G. Dean](#): *It's inherent within the system. Very briefly the argument is something like this: without TCAS you would have 5 mid-air collisions in a period of time and an airspace. With TCAS, you would have 2 collisions left. However, one of those 2 collisions has been created by TCAS. You have still considerably improved the overall safety, but one of the collisions has been induced by the system.*

[P. Johnson](#): *Presumably, the Überlingen accident falls in that category.*

[G. Dean](#): *Absolutely! You can still argue that there was a weakness in the logic, I would call it a flaw, in that...*

In comparison with OSS, you have to pay a fee to access the RTCA documents. While the fee is not more than a few hundred dollars, trivial for an organization, it is significant for individuals. This money is used for funding the RTCA, so it raises the question of how to fund the people responsible for establishing the standard.

Feedback from users experience has been incorporated into upgrades of the system. Since the initial release, there have been 2 upgrades, and there could be another one.

Individual manufacturers add their own features, much like Red Hat or Novel do with Linux. However, these additions are not fed back to the RTCA; they are the manufacturer's selling points.

[Some of the questions](#) distributed to all of the authors are worth considering here:

- It's not just the motivation of developers that counts. Budget holders in airlines and ATM authorities still need to be motivated, even with "free" software, since there are associated personnel costs.
- About security symmetry⁴, the hardware can help anti-spoofing measures.
- Practically, collaboration is facilitated by teleconferences, web site, wiki, email and regular meetings. These meetings are a great help in understanding other people collaborating with the project.
- Build peer review by welcoming criticism, not taking it personally, and have good faith in people involved.
- Substantial effort was used to develop TCAS evaluation criteria and software tools, and this was worthwhile. How do you know you've improved something if you can't measure it? A couple of millions of dollars (1 or 2 percent of the budget for the development of TCAS) were spent just on developing the evaluation criteria.

⁴ In a private communication, in reply to a question of Garfield Dean, Prof. Brian Fitzgerald explained: «'Security Symmetry' is a reference to Ross Andersen's conjecture (discussed in *Perspectives on Free and Open Source Software*, The MIT Press, Cambridge, USA, June 2005) which proposes that open systems may be more prone to security attacks (because 'evil' crackers can see the code) but this is balanced by more opportunity to identify and fix potential security flaws in the first place (because 'good' hackers can also see the code). Breaking the security symmetry would be trying to shift the balance more towards realising the benefits, at the expense of incurring the risks.»

- We need to accept many different motivations for working on OSS. Even those trying to break it, probably help make it more robust.
- There have been competing demands. There are substantial differences between US operations and European operations. There must be an arbitrator for competing demands, who arbitrates the final compromises in setting the parameters of the system. Ultimately, who is it?

Finally, we can speculate where OSS could be successfully introduced in ATM. Perhaps we should start using OSS only in non safety-critical systems? At least initially, that could prove easier to do. Clear lines of responsibility are required. This does not mean that safety critical software should not be open to review. Who will be in and run the steering committee?

Initial suggestions could easily include safety and performance analysis tools, real-time and fast-time simulators. These types of tools are not safety-critical, and would be useful and widely required by the community. Why shouldn't we have a world-wide community producing professional simulators? Amateur-built flight simulators have been developed by communities, so why not pitch in and use that work in professional simulators?

Why not having an open source ESCAPE or an open source RAMS? These are our in-house proprietary simulators at EUROCONTROL.

15.2. DISCUSSION

from 29'15" to 36'48" (07'33")

[H. Lueders](#): *How do you achieve TCAS interoperability?*

[G. Dean](#): *Through specific standards. Normally one airplane reacts before the other and communicates its intentions. The first aircraft that sends a message is the one that will lock things in. In the very rare occasion where there's a synchronous choice of resolution advisories, then the aircraft with the lowest Mode-S address is given priority and forces the other one to reverse its choice (each aircraft in the world has a unique Mode-S address).*

[P. Johnson](#): *So the RTCA process obviously does work generally...*

[G. Dean](#): *That sort of standard setting process is somewhat useful in this context. I'm sure that ATM regulators are going to require standards to be set, but the question is how to make that process more open to the man on the street.*

[P. Johnson](#): *The ITF has a very practical model.*

[O. Robert](#): *There's a big difference between the ITF and the ICAO or the RTCA. The last two organizations are trying to do things before there is any implementation. The ITF tries to validate implementations, because to have an internet standard, you have to have two different interoperable implementations. It's completely the opposite process.*

[G. Dean](#): *In practice, the implementation and the standards go hand in hand. Manufacturers start to develop the kit and put in some provisional algorithms to see how things were working and in the meantime people were developing alternative algorithms.*

[R. Schreiner](#): *I think it is more the OMG (Object Management Group) style.*

[P. Kappertz](#): *It not so clear why this approach is called "open-source" software. It looks like a perfect example in which you really can define standards that are applied, work, and are interoperable.*

[G. Dean](#): *The standards are at an exceptionally low level, they are really getting down to the engineering details . It interesting to ask "To what extent can you push the standards down to the details?" "Can you set an open source as a standard?" That's perhaps the challenge.*

[M. Bourgois](#): *On a social point of view, there's a strong similarity with the process of open source.*

[J. Feller](#): *Open Source projects tends to flourish when they are clear standards. Take for example the W3C, or the Apache Foundation's programs like Tomcat or Jakarta.*

Another thing I want to say is about your question of whether open source software is appropriate fo safety-critical systems. The answer is probably the same if you ask the question whether proprietary software is appropriate for safety-critical systems. You have criteria in mind when evaluating proprietary systems, and they are no more or less stringent for open source.

[G. Dean](#): *Fair comment!*

15.3. SALIENT POINTS

- A Traffic Alert and Collision Avoidance System (TCAS) must interoperate with other units.
- Since TCAS was a project with global scope, the FAA had to develop international standards from the onset.
- A high-level set of performance standards was created by the International Civil Aviation Organization (ICAO).
- The detail work was done by the Radio and Technical Committee of America (RTCA).
- Initially, the system was fluid, with rapid incorporation of changes. Once the system was implemented, however, incorporating changes became much more difficult.
- The individuals participating in this process were getting paid. The various organizations sending delegates had to cover these costs, of course, but the individuals involved received monetary compensation for their efforts.
- Although TCAS is a safety system, it is not a safety-critical system.
- Fee money is used for funding the RTCA.
- Individual manufacturers add their own features, but these additions are not fed back to the RTCA, they are the manufacturer's selling points.
- Substantial effort was used to develop TCAS evaluation criteria and software tools, but this was worthwhile. How do you know you've improved something if you can't measure it?
- Initial suggestions for OSS ATM software could easily include safety and performance analysis tools, real-time and fast-time simulators.
- There's a big difference between the ITF and the ICAO or the RTCA. The last two organizations are trying to do things before there is any implementation. The ITF tries to validate implementations.
- Open Source projects tend to flourish when they are clear standards.

16. QUALITY IMPROVEMENT AND RELEASE MANAGEMENT

by [Martin Michlmayr](#), University of Cambridge

16.1. OVERVIEW

Quality is a hard thing to define, because while most people can perceive the difference in quality between two objects, objectively describing it is difficult. Thus while we all agree that quality exists, finding a measure or metric for quality is a daunting task.

One of the dictionary definitions for quality is "fitness for purpose". This is the basis of a software quality metric: comparing the software to its specification. However, this presupposes that the specification itself is complete and of good quality. In addition, most open source projects don't have a formal, written specification to compare against. Thus, this isn't very useful for our purposes.

We're forced to recognize that quality is not a single thing. Attributes of quality include efficiency, reliability, usability, extendibility, portability, reusability, maintainability. Perspectives count as well. The user's perception of software quality will be different from the developer's perspective. The source code may be beautiful, but the program may be hard for the user to master.

So for software quality assurance, we could say that the code must do what it's supposed to do. We can also compare our code with that of the industry, and see if it works as well as theirs. Some OSS software has reached industry standard levels, like Linux in the operating system arena, while other OSS efforts lag behind.

Often during development, under time pressure, an attitude develops, "let's just get the software working, we'll worry about quality when we have time." Unfortunately, there never is enough time, and if quality is not built in from the beginning, the final result can be a hopeless mess that needs rewriting to become quality code. And there's never enough time for another re-write...

[F. Gasperoni](#): *Quality cannot be added on later, but it can be reverse engineered from quality software. Certification, for example, examines code to see that quality issues are met.*

The ISO definition is "all planned and systematic activities to ensure quality". This can seem contradictory with the volunteer nature of most OSS development. Many volunteer developers work when they have time, and their work is often not very "planned". So the fundamental question becomes, can you have assured high levels of quality when the developers are volunteers?

Since we've often mentioned "the community" in today's discussions, we should perhaps take a moment to ask ourselves, "who is that community?"

An example scenario: a graduate student at a university develops software while earning his degree. He finishes, and leaves the university. People in the university say to themselves, "Hey, let's open source this software. We'll put it up on a web site, and 'the community' will come and take over its maintenance." Companies can do the same with products they're no longer selling. Somehow, 'the community' will come and take care of this code.

But does this community exist?

Traditionally, in most OSS development, the work is done by volunteers in a distributed way. But more and more, open source is made by paid people who are collocated.

In EUROCONTROL's position, most work is done by paid personnel. But if amateurs build flight simulators and model train controllers, couldn't you get them to come into your community?

How could you motivate this community? Maybe the simulator builders could have closer access to aircraft profile data in exchange for their efforts. Or imagine that after some future air accident, the data and programs were made available to the community, with the challenge of finding a way to change the software to avoid this type of accident in the future.

This community won't come automatically. Probably, EUROCONTROL will be using a distributed, paid workers approach to development, but could also attract outside volunteers. You would have to be creative about finding a challenge.

Open source can coordinate in many different ways, from open groups where any developer may make changes, to restricted access for code commits. It is quite acceptable that your projects are managed and organized differently from other open source projects. There is no "the" or just one open source model.

In some projects everyone is allowed to make changes. In other OSS projects, you need to have permission to make changes.

In ATM you need test suite, but in most OSS application today, there is no test suite. But it is perfectly acceptable that the ATM community works in a different way to something that does not have safety concerns.

There is an underlying assumption behind most OSS that quality is high because there are thousands of eyeballs looking at the code, but this is simply not true in most cases. While Apache may be high-profile, top quality code, a careful look at Sourceforge will turn up thousands of dead projects with poor quality code, and no community around them. Since it is so simple to set up open source projects, many are created without sufficient resources.

Open source project management can be a problem, especially in large organizations like Debian, where release dates slipped by months, then years.

Academic work mostly concentrates on successful open source projects. Very little attention has been given to the failures of the problems. So we did a series of interviews on what problems were encountered in open source projects. We identified three areas of interest:

Leadership: there isn't one single open source model. Some are organized as a team and vote, some have an authoritarian style. For ATM, it would have to be a very tightly controlled model.

Release cycles: there is always something to improve. There are different release processes.

Company involvement: Since a company can be more rigorous, it can contribute documentation and test suites.

In conclusion, you must first identify where the community comes from. In your case, it would mostly be paid workers, but if you want to attract volunteers, you have to find incentives. If you want to use third party developed OSS, you will have to verify quality of the code, just as you must do with proprietary code. Finally, there is some academic work available on quality in OSS, so you can find some guidelines for high quality OSS.

16.2. DISCUSSION

from 23'00" to 29'35" (6'35")

[J. Feller](#): *We had insights from Debian. What about the BSD approach?*

[O. Robert](#): *Speaking about FreeBSD, we try to be less political. We do try to have some quality assurance.*

[M. Michlmayr](#): *Debian and BSD are not better or worst, they are just different with different beliefs. That is very important when you establish a new community, it's very important to make very clear what your community is about. You also have to think about the growth of your community, to be able to get rid of time-wasters and keep your best developers. Especially in this area, where you need safety, it has to be tightly controlled, but yet open enough to allow people to get involved.*

[R. Schreiner](#): *Comparing Linux and BSD, the objectives are a bit different. Linux developers write drivers for the latest hardware and that might generate instability. On the other hand, BSD does not want to support new features, so has a more stable kernel. It's a different approach. Therefore, the air traffic control industry would maybe lean towards BSD.*

[M. Michlmayr](#): *When you have a project, you must choose the most appropriate culture. BSD is probably better for mission-critical projects. Many BSD systems have uptimes of almost 500 days. Nevertheless, many Linux-oriented projects are very strict.*

[J. Feller](#): *The distributed/collocated by paid/volunteer matrix is a very valuable planning tool. If you concentrate on collocated non-volunteers -- another word for that is "employees" -- institutionalising a process of sharing and learning will enable you to then go to the distributed non-volunteers, which is where you get to partner with other organizations.*

If you actually had EUROCONTROL and EUROCONTROL's equivalents in other regions and the major manufacturers in on the game, you'd had no trouble at all attracting the efforts of people who want to learn and to contribute, and be part of all that, because suddenly you have this critical mass of industrial and government community that represents an incentive for young professionals. There's a big difference between having the tools available to facilitate collaboration and giving people a reason to do it.

16.3. SALIENT POINTS

- Attributes of quality include efficiency, reliability, usability, extendibility, portability, reusability, maintainability.
- Quality cannot be added on later, but it can be reverse engineered from quality software.
- If you actually had EUROCONTROL and EUROCONTROL's equivalents in other regions and the major manufacturers in on the game, you'd had no trouble at all attracting the efforts of competent people, because suddenly you have this critical mass of industrial and government community that represents an incentive for young professionals.
- There's a big difference between having the tools available to facilitate collaboration and giving people a reason to do it.
- You must first identify where the community comes from. In your case, it would mostly be paid workers, but if you want to attract volunteers, you have to find incentives.
- If you want to use third party developed OSS, you will have to verify quality of the code, just as you must do with proprietary code.

17. OSS IN SECONDARY SOFTWARE SECTOR, VOICE OF INDUSTRY

by [John O' Flaherty](#), CALIBRE

17.1. OVERVIEW

OSS is a commercial reality, like it or not. In any case, EUROCONTROL will have to deal with it.

In a feedback session during a conference last year, we gathered a list of positives, negatives, and emotional reactions to open source. The top half-dozen from a ranked list:

1. People issues: culture, social.
2. Commercial realities.
3. Technology.
4. Community.
5. Product: quality, features
6. Legal: despite all the discussion around IP, it's not really that much of a problem. You just have to deal with it. Putative legal uncertainties concerning OSS are mostly marketing material, just Fear, Uncertainty, and Doubt (FUD) from proprietary software houses to attempt to discredit OSS.

Now, for EUROCONTROL. We've been asking the question all day, "should EUROCONTROL go open source?" Maybe we should rephrase the question as "Why aren't we doing open source? Why should it be proprietary?" You're a publicly-funded institution, running in a sector specifically intended to promote collaboration.

Let's first look at the *technical infrastructure*. Positive aspects include transparency, since you can see the source and assess its quality. You gain increased control from this transparency, and you can start to articulate the standards, at the lowest level. Software reuse is the big payoff. Hand in hand with open source are open standards. Negative issues include reliability. Can you use this for safety-critical systems, particularly in the ATM domain? How do you find the solution you require, out of thousands of active or moribund open source projects?

The *social infrastructure* is probably where the positive difference would be greatest. This is where you have to ask the question why not open source? If EUROCONTROL stipulated that in the absence of compelling reasons not to open the source code, all code should be open source, the perspectives for cooperation would widen. For example, in the AdaCore case, a single stipulation in the base contract created the fertile open source environment where AdaCore flourished.

Shared knowledge. How would the member states react? Would openness at EUROCONTROL stimulate openness for member states? Shared R&D costs are another positive aspect.

Critical negative issues in social infrastructure are again community. It's not easy to create a community. It requires resources and effort. When I visited the OpenATC web site, the thing that struck me was their mention that they had no resources. They couldn't succeed. Building community requires time, money, and support. When the resources are there, people can meet and share. They want to be part of the fun of it. Then you get the really great programs.

You need open source skills, and open source skills are definitely different. You need to get those skills. Either by training or by hiring, you need to get the skills to work with the community.

A more ambiguous aspect is that open source is global. It's not just European. The US, China, India - everybody will be in on the act. Your members may have issues with that. To have a community, you need to build trust and like any trust is about taking and giving, and to make it fun for the people involved.

J-P. Florent: *If you improve the ATC in China, it is for the benefit of the passengers and the airlines.*

J. O'Flaherty: *So it does fit your mission.*

M. Bourgois: *We have no restraint in co-operating with anybody, because eventually it will benefit to the passengers, the ultimate clients of the system.*

B. von Erlach: *We are mainly a European organization, but since the ATM is a worldwide system, the benefit is also worldwide.*

M. Bourgois: *We will not focus specifically on China. If they are benefits and they use or abuse it, that is good.*

M. Michlmayr: *You can also have a support model while you have the expertise.*

R. Schreiner: *We develop an Open Source security system, and 40% of the investment comes from China and India. They really do like OSS from the US and from Europe.*

M. Bourgois: *We do work for ATM not for airlines. Even Chinese airlines flying European airspace will bring benefits for the ANSPs in Europe. The game could change once they start to produce ATM software.*

G. Dean: *They are likely to do that in the future, given that China is rapidly expanding its airline fleet.*

J. O'Flaherty: *So China is a questionable issue, probably positive.*

In terms of legal infrastructure, the biggest issue is: Who do I sue, and who is going to sue me? *Legal aspects* boil down to those two questions, and the answer is unclear. But you can't just be neutral; you're going to get OSS anyway. As the PHILIPS presentation illustrated, open software will enter the enterprise in any case, you've got to figure out how to deal with it.

On the positive side, given EUROCONTROL's role, there is a strong motivation to be open and collaborative, and it does foster open standards. On the negative side, there's the ownership mind-set. Patents, used as a competitive marketing tool, are a bad thing. Spurious patents, rather than patents themselves are the issue. The biggest problem on the legal side is the lack of understanding of the issues. Companies dealing with open source have support centres to advice developers on legal issues. Perhaps EUROCONTROL could provide that support for ATM developers. Bigger companies can afford it, but the little ones can't.

The *corporate environment* for EUROCONTROL with open source would be a more cooperative, more flexible environment, offering greater job satisfaction. To develop great software, get great developers. The open source meritocracy fosters excellence, attracts great developers, and most importantly, helps retain them. On the negative side, there is a resistance to change. The tension between open source meritocracy and business profit motive will probably be tilted toward meritocracy, given EUROCONTROL's role. But high-level expertise, at high decision making levels, is another problem. If a member state decision-maker protests that a proprietary firm in their country will shut down if you move to open source, you'll have to justify their job losses.

Finally, in all successful open source projects, you always find a person, a champion for the cause. The presence or absence of a driving force champion is essential. If you have the champion, and you support him, the community will follow.

What would open source bring to EUROCONTROL? It would allow smaller companies to enter the field. Submitting good code could result in service contracts. But this applies to Russian and Chinese firms as well. Risk remains present, but can be managed. In terms of time to market, the impact is unclear. You can't just rush safety-critical applications to market.

Business models are also unclear for open source. Proprietary companies may have an advantage here: they've years of experience; they know how to respond to and win tenders; they can and will lobby officials. New OSS based companies need to learn that. Probably the best solution is hybrid, a mixture of open source and proprietary software. EUROCONTROL is in a position to provide the leadership for such an environment.

How do you sustain the open source? Someone has to fund the champions and the community. This takes time and money. You must have a hybrid fallback position: the real world is not open source or proprietary, but rather a mix of both.

In terms of acceptance, open source could invigorate, offer an open, progressive position, and provide good public relations. A downside is that you could be rejected by the libre community, or by the customers. Current vendors probably have something to lose with open source.

A suggestion could be to use open source as a focal point, a direction, a mission to go for, as Kennedy said about going to the moon. EUROCONTROL should choose OSS. Not because it is free or easy, but to succeed in it will create a more valuable industry, because that goal will serve to organize and measure the best of the ATM sector's energies and skills, because that challenge is one that EUROCONTROL is willing to accept, one it is not willing to postpone, and one which it intends to win.

17.2. DISCUSSION

from 17'45" to 30'00" (12'15")

M. Bourgois: *Before anyone leaves, I would like to thank the people that have been involved today, especially the speakers who took time to share their thoughts, the other participants who brought additional insight, and Jean-Luc Hardy for the organization.*

H. Lueders: *There's a lot of knowledge out there. There is high need to look more closely to the skill requirements that was just mentioned. We should have a special event to bring the experts together and address these hard questions.*

M. Bourgois: *That's a very good recommendation to CALIBRE and CALIBRATION. By the way, I want to thank them especially as well for co-organizing this event.*

J. O'Flaherty: *We actually did a workshop at Genova in July on Open Source education. There are a number of Master programs that teach OSS, but not at the industry level.*

J. S. Swierstra: *Thank you very much for organizing this. It was extremely useful for us to see the legal possibilities, we had the impression that this would have been quite constraining.*

In our division, we are looking at the transition from where we are today, to the dream system of the future. This means a paradigm shift in the introduction of automation in ATC systems. There is very little automation in the system today, and the system in the future is supposed to do it all by itself.

It's a very small community. There are three main players in this community, but they are not here today, which is an interesting signal. To achieve a minimum level interoperability, we must more or less force commonality on the functionality and technical solutions that these main players will have to provide to the users, the ANSPs. Even if they don't voluntarily want to go to some kind of open source in our small community, if we want to create advances for ATM systems, we, our division in Brussels headquarters, are absolutely convinced that we have to do something here.

We have to come to some kind of reasonable cut between what is really profit-making for these companies, where their expertise lies, and what is key point for us, which is to assure that we have interoperability. We don't want to say, "everything we do must be open sourced," that's not the issue. But there are certain elements which drive the essence of interoperability not only between ground systems, but also between ground and air systems as well.

There's a big scope for us. As you indicated, we need some organizations or persons to drive this, but there is a lot of wind against this. The fact that the three main players are not here today is a good indication of this. We have been discussing this for several years now, and the individuals from industry we talk to try to convince us that this is a very bad idea. This will not stop us from saying, "that is your vision, but this is the way we must go."

[M. Bourgois](#): I would like to put your comments about the behaviour of the industry in context. The middleware standardization problems do illustrate that it is not easy to get the industry to take up open approaches or shared standards. However, for today's event, we did not push them to come, because we had limited resources. The target here was to confront internal projects of EUROCONTROL with expertise in open source. That was the stage of this discussion, and one should not read too much into the absence of industry representatives in today's meeting. A few actually are present today, and there's no sign at all of boycott by industry. But we do know they're not generally sympathetic to an environment which could create more competition.

[J. O'Flaherty](#): It's not really an either/or situation. It would probably be to the benefit of the three major players if the open source approach is adopted. The PHILIPS presentation illustrated that nicely: it's a reality. They've just got to deal with it.

Also, if you open up a community around projects, you can attract other people. And from a European point of view, open source gives an opportunity for brilliant academics to contribute to mainstream commercial software. The community could get a lot bigger, with other players than you would expect.

[P. Johnson](#): Can I suggest that the ANSP cannot wait for suppliers, because open is generally for the benefits of customers to the expense of suppliers who could be against it.

[J. S. Swierstra](#): With the privatisation, the roles of clients and suppliers in ATM are perhaps less clear. The ANSPs are not clients only. Some of them are becoming extremely sensitive to IPR issues. It is a very strange situation: one the one side, the ANSPs are our boss, on the other side they are bidding on the contracts that we are putting out.

[H. Lueders](#): The interoperability issue is every day on the agenda of the European Union, and not only in Brussels, but also at the level of the Member States. CompTia is everywhere involved. At the moment, we wait for the new communication by the Commission on interoperability. In the first quarter of next year, there will be the launch of the interoperability centre. I could add and add to that list, to say that it is high on the agenda. And the same for the IPR issues perhaps even more important at a longer term.

[J. Feller](#): *Just one final word of encouragement. Earlier today, I was asking Franco about the existence of competitor for their core product, the GNAT. What I was hearing was that there are people out there who build derivative works and then sell their expertise, but there is no one competing with them on their core product. I do not see that as a negative or surprising thing, because the core asset of AdaCore is a unique knowledge of their particular space.*

Now, open source started in well defined horizontal spaces. What we see now is a slow emergence of vertical spaces, like open source ERPs for example. The barrier that prevents open source of penetrating in these vertical spaces is a deep knowledge of the vertical spaces.

So, for EUROCONTROL, the one asset that you bring to the table is that unique knowledge. The same is true with AdaCore. And that is the key encouragement: if you want open source to work, you are the only people that could make it work.

[M. Bourgois](#): *Thank you very much, and I would like to conclude with this excellent statement.*

17.3. SALIENT POINTS

- OSS is a commercial reality, like it or not. In any case, EUROCONTROL will have to deal with it.
- Despite all the discussion around IP, it's not really that much of a problem. You just have to deal with it.
- EUROCONTROL is a publicly-funded institution, running in a sector specifically intended to promote collaboration.
- Software reuse is the big payoff. Hand in hand with open source are open standards.
- For EUROCONTROL, benefiting air passengers in China will benefit air traffic worldwide, and is part of the mandate. On the other hand, manufacturers in the ATM sector may have difficulty competing with Chinese or Indian firms also using European-developed OSS.
- If EUROCONTROL stipulated that in the absence of compelling reasons not to open the source code, all code should be open source, the perspectives for cooperation would widen.
- To develop great software, get great developers. The open source meritocracy fosters excellence, attracts great developers, and most importantly, helps retain them.
- Building community requires time, money, and support.
- The presence or absence of a driving force champion is essential.
- You can't just rush safety-critical applications to market.
- You must have a hybrid fall-back position: the real world is not open source or proprietary, but rather a mix of both.
- Maybe the best approach would be to use open source as a focal point, a direction, a mission to go for.

18. PRELIMINARY DEBRIEFING: LESSONS LEARNED

by [Marc Bourgois](#), EUROCONTROL, 15th Dec 2005

The challenging and at times very animated debate has lead us to formulate the following five insights:

- OSS is a reality for every business.
- Licenses are not the barrier.
- Communities which are successful.
- Creating value with OSS.
- No OSS experience in safety-critical domains.

18.1. OSS IS A REALITY FOR EVERY BUSINESS

The first and probably most fundamental insight came when asking the question “why bother?”. Why should EUROCONTROL bother about OSS? The answer is very simple and very generic, i.e. it applies to all organizations which are heavily engaged in software development: any organization or business is confronted with OSS, whether it set out to do so voluntarily or not. Any software being procured runs a high risk of including some OSS.

Any software being released runs the risk of being included in some OSS further down the line. And if it were not for one’s own suppliers or customers, one’s own software developers (a notoriously independent and free-thinking breed) are likely to rely on some OSS.

Depending on the specific licenses that go with the OSS, as we will see next, the risks can entail important commercial consequences. In one case, PHILIPS had to distribute tens of thousands of CDs. In another case had no realistic option but to release its software as OSS because of the inadvertent inclusion of OSS.

What can be done to contain the risks associated with inadvertent inclusion of OSS? Three measures are possible: (i) scan the software with specialized tools for OSS detection and subsequently eliminate its reliance on OSS; (ii) strategically decide on which software is commercially interesting and subsequently engineer an architecture that allows the separation of proprietary software and OSS; (iii) define an in-house policy (which should include the previous two measures) to increase awareness.

It became very apparent during the discussions at the roundtable that there is a lot of confusion in the ATM domain (and we would suspect in any domain only recently confronted with the open source paradigm) about what exactly qualifies as OSS. We had already identified the problem during our fieldwork preceding the roundtable. Most importantly, the concept of freeware is often mistaken to be identical to OSS; not in the least because both are often accompanied by very active user and development communities. One cannot quote enough the famous analogy of Richard Stallman, one of the gurus of the open source paradigm, that “free as in freedom, not as in free beer”. Terminology confusion has been plaguing the OSS community itself for as long as it existed as well. No wonder that, given the plethora of expressions – OSS, Libre Software, Free Software, FLOSS – the legal experts starting out their exposes at the roundtable with a set of definitions, which takes us to the second lesson learned:

18.2. LICENSES ARE NOT THE BARRIER

A facilitated brainstorming session at a recent CALIBRE conference led to the prioritisation of issues that have to be dealt with in order for OSS to become a mainstream paradigm. Legal issues ended at the sixth and last place; clearly licensing and related issues are not expected to be the showstopper. Our roundtable discussion strongly confirmed this appreciation.

There exist a bewildering number of open source licenses (200 by one count). The large number of such licenses is both a curse and a blessing. A curse because it leads to confusion and scares away non-legal people. A blessing because the license which would fit the requirements of your project probably already exists and merely needs to be adopted.

While admitting that the IPR issues surrounding OSS are complicated (even legal experts do not necessarily agree on all the details, as we could witness at our roundtable), the multitude of licenses can be reduced to three broad categories: (i) Free-for-All open source, (ii) Keep-Open open source and (iii) the viral Share-Alike open source. The categories are based on two basic obligations for the user of the license: the need to credit the origin and/or the obligation to share modifications and possibly extensions.

In fact, the EUROCONTROL agency has already released one piece of software under an OSS of the second category. It concerned an ADA utility, which was re-integrated in the core OSS release on request of external users. Admittedly this piece of software cannot be considered an ATM application; nevertheless it offered a first opportunity for the legal service to get acquainted to the issues and to management to take a stance.

So the solution to the license issue consists in dealing with the issue proactively by formulating the exact strategic / commercial requirements of the project. This requires an early involvement of legal experts in order to reduce the risks and identify the appropriate solution.

18.3. COMMUNITIES THAT ARE SUCCESSFUL

Probably the most quoted strength of OSS projects is their (supposedly) large and responsive developer community. Proponents of the OSS paradigm attribute the quality and long-lived successes of OSS software projects to the abundant and free contributions of large distributed teams of volunteer developers.

As has been shown, many of these claims only hold for a very small subset of OSS projects; neither is the verdict on OSS quality easy or uniformly positive. The research on OSS has only just begun to investigate more in depth the relationship between characteristics of the OSS community and the resulting software quality. Indications are that the set of tools being used in the development process, the motivation of the developers are all factors which may positively correlate with quality.

Currently a trend can be observed of ever more developers contributing to OSS during their normal working time and with the full endorsement of their employers. This is a reflection of the maturity OSS is reaching in certain areas of software development; mainly infrastructure-related or so-called horizontal areas such as. The original assertion that large numbers of developers contributing for free are essential to successful OSS is clearly being countered by this evolution. The current assertion is more subtle and considers the quality of contributions (which by definition is relatively high amongst professionals) to be more significant than the amount of contributors. This holds promise for a small vertical domain such as ATM which would always have difficulty in attracting the larger base of "hobby" developers working from their homes in their free-time. Nevertheless the roundtable did go into a small excursion to explore how even an arcane domain such as ATM could tap into and motivate the larger community of OSS developers. One of the

more promising ideas mentioned concerned creating an ATM simulator (much like flight simulators).

As mentioned before, several ATM experts presented early OSS experiences during our roundtable. Interestingly, two of these were described as “failed” OSS projects. One was a portal / forum created by EUROCONTROL in conjunction with another major ATM research centre. The portal was to host free software put at the disposal of the larger ATM research community for use, discussion and evolution. Surprisingly, although several research tools were uploaded, little or no activity took place on the forum. This remained unexplained until the OSS experts enlightened us on two “soft” issues determining the success of an OSS community: motivation and championship.

On the one hand it seems to be essential to involve the community early in order to avoid a “not-invented-here” reaction when they are confronted with a “fait accompli”. Rather than submitting finished contributions, the relationship between the contributing organization and the OSS community should be based on the golden rule of “release early, release often”. On the other hand, all new initiatives require an energetic champion, or “benevolent dictator”, to guide the often chaotic evolution of a self-motivated development community. Both lessons should be taken serious when selecting pilot projects for OSS adoption.

18.4. CREATING VALUE WITH OSS

Let us come back to the second OSS example from the ATM domain which was presented as a “failure”. It concerns an application called AudioLan. AudioLan, which implements Voice-over-IP for ground-ground communications between controllers [to be checked whether the claim of air-ground can be substantiated], replacing the analogue telephone switches, was originally developed on behalf of EUROCONTROL. The early version was quickly adopted by over 10 European sites, with 500+ positions.

For the second time, the OSS experts at the roundtable came up with a plausible explanation. Based on their wider experience, allowing them to draw parallels, they strongly argued that this effect is being witnessed whenever the piece of OSS is crucial to differentiate between the products of the industrial players. For an OSS effort to be successfully adopted by established industrial competitors, the application should not be central to the competitive advantage of their respective products.

In such situation fierce competitors may behave, quite counterintuitively, in an altruistic manner, contributing substantially to the standardization of the OSS functionality. A good example from a horizontal domain is the X-Windows implementation, which flourished under the sponsorship of an industry consortium because the application was not seen as providing any valuable product differentiation, but it was seen as a necessary enabler for user-required interoperability. X-Windows may be the best known such example of a reference implementation which has become the unique implementation, but it is by now means the only example. We believe that this insight holds promises for successful adoption of OSS in the ATM domain.

As pointed out in a CALIBRE interview before the roundtable, the question came up whether the ATM supplier industry is not already a service industry, and it would be fooling itself when behaving as if its major income stems from basic ATM applications and not from system integration, maintenance and other service activities.

18.5. NO OSS EXPERIENCE IN SAFETY-CRITICAL DOMAINS

Finally, and here we can be very short, we noted the lack of OSS penetration for safety-critical applications. No examples could be found in the literature; neither could any be recalled by the OSS expert panel at the roundtable. It is worth noting however that there is ample anecdotal evidence of OSS communities which are very responsive to critical bug reports, much more responsive than what would usually be specified in contracts with proprietary software providers of safety-critical applications.

Does this mean that the ATM domains cannot benefit from the potential of OSS? No, it merely means that the safety-critical applications in ATM should not be the first to be explored. But then again, there are very many non safety-critical components in the overall ATM system, so plenty of opportunities to build experience with OSS exist. In fact, one ATM expert at the roundtable ably, but provokingly argued that ATM applications are not safety-critical at all, because the traffic is constantly kept conflict free by definition, offering several minutes of reaction time for the humans in the system to deal with outages of automated components. The argument continues to identify the avionics components as the truly safety-critical parts. With the unavoidable increased air-ground integration of future evolutions of the ATM system the borders of what is a safety-critical application and what is not are definitely going to blur further.

19. ROUNDTABLE PROCESS

Since the round table could trigger similar initiatives for the adoption of OSS in other industrial domains, here are the milestones of its organization, with the names of the persons in charge.

- Proposal of an ATM/OSS cross-fertilization event:
Jean-Luc Hardy [JLH], EUROCONTROL
- Endorsement by EUROCONTROL,
as a roundtable in parallel of the INO workshop of Dec 2005:
Marc Bourgois [MB], EUROCONTROL
- Endorsement by CALIBRE:
Brian Fitzgerald [BF], University of Limerick, Ireland
- Programme:
JLH
- Invitation to OSS experts:
BF and Andrea Deverell [AD], University of Limerick, Ireland
- Invitation to ATM experts:
JLH
- Texts and slides of presentations:
Elicitation: JLH
Authors: cf. programme
- Logistics for participants
Registration: Martina Jurgens [MJ], EUROCONTROL
Venue: MJ and JLH
Printed flyer: AD
Printed pre-proceedings: JLH
- Voice recording
Master: Herve Bechtel, Gilles Chedozeau, EUROCONTROL
Backup: John Seifarth [JS], Words-and-Wires
- Moderation of the roundtable:
MB
- Participation to the debate:
cf. Programme and Participation
- Website with proceedings
Initiation: JS and Colette Uytterhoeven [CU], Words-and-Wires
Software and IT support: JS
Specifications, navigation, HMI: JLH
Transcriptions: JS, CU, JLH, and JOF
Photos: John O'Flaherty [JOF], CALIBRE
Sound filtering: JLH
- Report (this document)
Editors: MB, JLH, JO, and JS
Logistics: JLH and Joyce Roelofsen, EUROCONTROL

20. SHORT BIOGRAPHIES

Carlos Garcia Avello, EUROCONTROL, guest speaker



Carlos has an engineering degree from Université Catholique de Louvain in Mécanique Mathématique. He has experience in the development of ATM simulation components and Decision support tools. His present activities are in Common Trajectory Prediction, decision support tools to facilitate Continuous Descent Approaches and RNAV routes into TMA and in the field of traffic complexity prediction for Multi Sector Planning.

Marc Bourgois, EUROCONTROL, co-organizer



Marc is deputy manager of the Innovative Research Area at the EUROCONTROL Experimental Centre, focusing mainly on emergent technologies and collaborations with universities. In June 2004, Marc joined the EEC from EUROCONTROL HQ where he has been working for 8 years in the European ATM Programme, most recently as the manager of a small unit responsible for business planning, continuous improvement and people strategy. These activities were a logical extension of his role as advisor to the Senior Director, which he performed for 4 years. Marc's first job with EUROCONTROL centered on systems architecture. Marc Bourgois holds a polytechnics degree from the Faculty of Applied Sciences of the University of Louvain and a master degree in artificial intelligence. He started working as a software engineer on factory automation and production scheduling with Siemens Corporate Research in Bruges. Later he transferred to the European Computer-Industry Research Centre, an international joint venture of Siemens, ICL and Bull based in Munich. As a member of the research staff he worked on logic programming, distributed systems and agents in the context of several EC research actions.

Garfield Dean, EUROCONTROL, guest speaker



Garfield has been researching in Air Traffic Management (ATM) engineering for 20 years. Initially examining potential applications of artificial intelligence in all aspects of ATM, for the past 10 years he has been analysing ACAS data and providing feedback to users and advice to policy makers. He is a regular user of and believer in OSS, in particular Linux and The GIMP; for example he has prepared his presentation for this roundtable using OpenOffice. He has succeeded in obtaining permission for a EUROCONTROL TCAS safety analysis tool to be distributed free of charge worldwide to aviation authorities.

Andrea Deverell, University of Limerick, co-organizer

Andrea is pursuing a PhD investigating the use of ambiguity as a means for increasing innovation and creativity in organisational contexts. She has over a decade of experience in higher education, both as lecturer and administrator, and her work has focused on effective collaboration and knowledge transfer between academia and industry. From 1999-2003, Andrea served as Technology Transfer Officer based in the Office of the Vice-President of Research, University of Limerick. In this role she facilitated over 35 research collaboration projects between academia and industry. Andrea is the Events and Industry Forum Coordinator for the CALIBRE project, where her work includes conducting research, coordinating CALIBRE's international workshop and conference series and facilitating the establishment of CALIBRATION, a European Industry Forum for OSS. Andrea is co-author of "[Assessing the Role of Open Source Software in the European Secondary Software Sector: A Voice from Industry](#)", which was awarded a Best Paper award at First International Conference on Open Source Systems ([OSS2005](#)). It was presented by John O' Flaherty during the roundtable.

Arnoud Engelfriet, PHILIPS, guest speaker

Arnoud (born 1974) is a European patent attorney working at the IP department (IP&S) of Royal PHILIPS Electronics. He drafted and implemented the PHILIPS policies on open source usage and release of PHILIPS software as open source. As secretary of PHILIPS' Open Source Advisory Board he coordinates the legal aspects of the use of OSS by PHILIPS. Arnoud is a regular speaker about open source. He has published several articles about legal risks of open source. He maintains the website www.lusmentis.com with over 250 articles on law and technology.

Joseph Feller, University College Cork, guest speaker

Jo is a Senior Lecturer in Business Information Systems, University College Cork, Ireland. He previously taught at the University of South Florida and at the Ringling School of Art and Design, and holds a PhD from National University of Ireland. His work on OSS includes two books (Understanding Open Source Software Development, 2001; Perspectives on Free and Open Source Software, 2005) and numerous journal and conference papers, book chapters, panels and tutorials, including 'A Framework Analysis of the Open Source Software Development Paradigm,' which was awarded Best Paper on Conference Theme at The 21st International Conference in Information Systems. Joseph was the lead organiser of the IEEE/ACM workshop series on Open Source Software Engineering (2001-2005), has served as guest editor for special issues on open source for Information Systems Journal, IEE Proceedings Software, Systèmes d'Information et Management and Software Process - Improvement and Practice, and has presented to a variety of industry and government workshops and briefings. Joseph is a senior researcher in the EU FP6 Co-ordination Action project CALIBRE (www.calibre.ie), in which he leads the dissemination and awareness work package and conducts research on OSS business models.

Brian Fitzgerald, University of Limerick, co-organizer

Brian holds an endowed professorship, the Frederick A Krehbiel II Chair in Innovation in Global Business & Technology, at the University of Limerick, Ireland, where he is also Research Fellow, Science Foundation Ireland Principal Investigator, and Research Leader for Global Software Development at Lero – the Irish Software Engineering Research Centre. He received his PhD from the University of London and his research interests lie primarily in the area of software development, a broad area which encompasses the use of development methods, globally-distributed software development, agile methods and OSS. His publications include seven books, and almost 100 papers in leading international journals and conferences in both the Information Systems and Software Engineering fields (including Communications of the ACM, IEEE Software, IEEE Computer, Software Process Improvement and Practice, Information Systems Journal, European Journal of Information Systems and MIS Quarterly).

Having worked in industry prior to taking up an academic position, he has more than 20 years experience in the software field. This experience was gained in a variety of companies (Citibank, eircom, IDS Computing, Ridge Tool Company) in a number of countries (Ireland, Belgium, Germany). He has also been successful in a number of research funding proposals from the European Commission, Science Foundation Ireland and Enterprise Ireland. Overall, these projects have received funding in excess of €15 million.

Jean-Dominique Frayssinoux, ATM consultant, guest speaker

Jean-Dominique has an engineering degree from the French National Civil Aviation School (Toulouse) and a Master degree in Electronic. He is ATM consultant, expert in radar/tracker technologies and a long expertise in software development and evaluation of radar trackers. He has a wide experience of ATM civil and military systems. He has also an experience as a pilot of private aircraft.

Franco Gasperoni, ATM consultant, guest speaker

Franco has an engineering degree from the Ecole des Mines de Paris, France and a PhD in Computer Science from New York University, USA. Franco has lectured and conducted research at New York University and the Ecole des Telecommunications (ENST), in Paris. While at the Ecole des Mines, Franco worked with Maurice Allais, the French Economics Nobel prize. With Cyrille Comar, Franco is co-founder and Managing Director of AdaCore (formerly known as ACT Europe), the company that develops, maintains, and offers commercial support for GNAT Pro, the Free Software development environment for Ada 83, Ada 95 and now Ada 2005. AdaCore is a 100% Free Software Company basing its business model on high-quality support and subscription-based pricing. AdaCore provides Ada solutions to customers in the air traffic management, avionics, military, railways, and space industries. Franco, who is a member of IEEE and the ACM has published over 25 papers.

Gilles Gawinowski, EUROCONTROL, guest speaker

Gilles has a M.Sc in Nuclear Chemistry and another M.Sc. in Computer Science. He also graduated in human factors. He has developed its professional background in the framework of innovative research applied to the Air Traffic Management. He has managed a variety of projects dealing with technical, human factors and operational issues.

Jean-Luc Hardy, EUROCONTROL, initiator and main organizer



Jean-Luc has academic degrees in Sciences of Education, Computer Sciences, and Business Administration from Belgian and US universities. Starting his career as a researcher of the Belgian FNRS at the University of Liège, he has published a book and about 40 papers in French and English. In 1988, he became a consultant for the R&D directorate of the European Commission at Brussels. He joined the EUROCONTROL Institute at Luxembourg in 1991, and has been involved in several training and awareness activities supporting the field of Air Traffic Management (ATM). Working at the EUROCONTROL Experimental Centre in France since 1996, he is presently assigned to the Innovative Research Area, being in charge of a study project to determine possible implications of the OSS paradigm in ATM.

Martin Michlmayr, University of Cambridge, guest speaker



Martin has been involved in various free software projects for about 10 years. He used to be the Volunteer Coordinator for the GNUstep Project and acted as Publicity Director for Linux International. In 2000, Martin joined the Debian Project, an association of roughly 1000 members working on a completely free operating system. Martin was elected Debian Project Leader in 2003 and after a successful term, he was reelected. In the two years as the leader of Debian, Martin represented the project and performed important organizational and coordination tasks within the project. Martin holds Master degrees in Philosophy, Psychology and Software Engineering, and started a PhD at the University of Cambridge in January 2004. His research focuses on quality management in free software projects.

Dr John J. O'Flaherty, CALIBRE, guest speaker



John is Project Manager of CALIBRE, and Technical Director of MAC, The National Microelectronics Applications Centre in Limerick, Ireland. John has 30 years experience in the software industry, both hands-on and development project management. He has been involved in managing, developing, evaluating and reviewing EU RTD projects for over 15 years. Just recently he was rapporteur for the Commission's consultation process on EU future OSS policy initiatives.

Delphine Prieur, INRIA, guest speaker



Delphine has a Degree in International Commerce and a Post Graduate Law degree from Université Paris V. In 1998, she started to work for the French National Institute for Computing Science and Control (INRIA) where she was first in charge of project administration including European coordination. Since 2001, she has been IP legal advisor at INRIA, specialized in the legal protection of software. She also advises University Paris Dauphine with regards to its IP strategy. At INRIA, she is in charge of the preparation of IP clauses for framework agreements with industrial parties such as France Télécom, Hitachi, Microsoft, Renault, etc. She is also in charge of the legal procedures of protection of INRIA results: patents, trademarks, software deposit, etc. She participated, on a legal point of view, to the creation of open source consortia, such as Objectweb, the open-source distributed middleware, and Scilab, the open source scientific software package for numerical computations.

Sip Swierstra, EUROCONTROL, guest speaker



Sip is responsible for the Centre of Expertise for Trajectory Prediction at EUROCONTROL Headquarters in Brussels, Belgium. After graduating as an electronics engineer he joined EUROCONTROL in 1973. He has been working on aircraft performance modelling, trajectory prediction and the application of these techniques in advanced ATM tools, in particular the Arrival Management concept “Zone of Convergence”

Burkhardt von Erlach, EUROCONTROL, guest speaker



Burkhardt is born in Berne/Switzerland in the early 60ies. He has a law degree from the Berne University. He made post graduate studies at McGill University in Montreal (LL.M in Air and Space Law). He worked at the Swiss Federal Department of Foreign Affairs in Berne (Transport Law). He joined EUROCONTROL in 1993, first at the Procurement and Special Agreements Section in Brussels and, since 2005, at the Experimental Centre at Brétigny as Head of the Finance Team. In both EUROCONTROL functions, he has been dealing with the intellectual property rights issues.

21. PARTICIPANTS

	Family Name	First Name	Company	Country	Status
1	ASARE	Bernard	LOCKHEED MARTIN	USA	(registered)
2	BARONNAT	Patrick	EUROCONTROL EEC	France	(registered)
3	BELOUARDY	Nabil	EUROCONTROL EEC-INO, student	Marocco	(registered)
4	BOURAOUJ	Jean-Léon Mehdi	IRIT	France	(registered)
5	BOURGOIS	Marc	EUROCONTROL EEC-INO	France	moderator
6	CELIKIN	Mete	EUROCONTROL HQ	Belgium	participant
7	CREBASSA	Philippe	DSNA	France	participant
8	DANG	Thong	University of Linkopings (NVIS)	Sweden	(registered)
9	DEAN	Garfield	EUROCONTROL EEC	France	quest speaker
10	DEN BESTEN	Matthijs	University of Paris Dauphine UPMC	France	participant
11	DEVERELL	Andrea	University of Limerick	Ireland	(co-organizer)
12	ENGELFRIET	Arnoud	PHILIPS	Netherlands	quest speaker
13	FELLER	Joseph	University College Cork	Ireland	quest speaker
14	FERCHAUD	Frédéric	EUROCONTROL EEC-INO, student	France	participant
15	FITZGERALD	Brian	University of Limerick, CALIBRE	Ireland	(co-organizer)
16	FLORENT	Jean-Pierre	EUROCONTROL EEC	France	participant
17	FRAYSSINOUX	Jean-Dominique	ATM Consultant	France	quest speaker
18	GARCIA AVELLO	Carlos	EUROCONTROL HQ	Belgium	quest speaker
19	GASPERONI	Franco	AdaCore	France	quest speaker
20	GAWINOWSKI	Gilles	EUROCONTROL EEC-INO	France	quest speaker
21	HARDY	Jean-Luc	EUROCONTROL EEC-INO	France	main organizer
22	JOHNSON	Paul	National Air Traffic Services Ltd	Great Britain	participant
23	JOYET	Pascal	Dassault Aviation	France	(registered)
24	KAPPERTZ	Peter	BARCO ORTHOGON	Germany	participant
25	LAPASSET	Laurent	NEOMETSYS	France	(registered)
26	LE LIDEC	Franck	THALES ATM	France	participant
27	LEPADATU	Catalin	EUROCONTROL HQ	Belgium	(registered)

28	LU	Wen-Chi	LAAS-CNRS	France	(registered)
29	LUEDERS	Hugo	CompTIA	Belgium	participant
30	MANCEL	Catherine	ENAC	France	(registered)
31	MICHLMAYR	Martin	University of Cambridge	Great Britain	quest speaker
32	MORA CAMINO	Félix	ENAC	France	(registered)
33	NEDELESCU	Liviu	LOCKHEED MARTIN	USA	(registered)
34	O'FLAHERTY	John	Microelectronics Applications Centre	Ireland	quest speaker
35	PAVET	Didier	DSNA	France	(registered)
36	PETERSON	Stephen	EUROCONTROL EEC-INO, student	Sweden	(registered)
37	POLLINA	Marc	M3 Systems	France	(registered)
38	POST	Wim	EUROCONTROL HQ	Belgium	(registered)
39	PRIEUR	Delphine	INRIA	France	quest speaker
40	ROBERT	Ollivier	EUROCONTROL EEC	France	participant
41	SCHREINER	Rudolf	Object Security Ltd	UK	participant
42	SEIFARTH	John	WORDS and WIRES	Belgium	participant
43	SILVA	Rodrigo	GISMEDIA	Portugal	(registered)
44	SIMONIN	Alexandre	ALTYS Technologies	France	(registered)
45	SORENSEN	Carolyn	ISA Software	France	(registered)
46	STERNE	John	Journalist of CALIBRE	Ireland	participant
47	SUCHKOV	Alexander	LOCKHEED MARTIN	USA	(registered)
48	SWIERSTRA	Sip	EUROCONTROL HQ	Belgium	quest speaker
49	TRUILLET	Philippe	IRIT UMR CNRS 5505	France	(registered)
50	VAN DER LINDEN	Franck	PHILIPS MEDICAL SYSTEMS	Netherlands	participant
51	VON ERLACH	Burkhart	EUROCONTROL EEC	France	quest speaker
52	WIMMER	Michael	BARCO ORTHOGON	Germany	(registered)

TRADUCTION EN LANGUE FRANÇAISE

RÉSUMÉ

À la mi-2005, le département de la recherche innovante du Centre expérimental d'EUROCONTROL a sollicité le concours du projet CALIBRE en vue d'organiser une table ronde sur le thème suivant : « Possibilités offertes par les logiciels libres dans le domaine de la gestion du trafic aérien (ATM) ».

Cette table ronde avait pour objectif de promouvoir une plus grande sensibilisation ainsi que de recueillir des faits et arguments en rapport avec quatre hypothèses générales concernant les implications possibles du concept de logiciel libre (*Open Source Software, OSS*) pour l'ATM. Pour y parvenir, la table ronde a été organisée de manière à promouvoir l'échange d'idées entre deux domaines d'expertise : celui de l'OSS et celui de l'ATM.

Étant donné que cette table ronde constituait le premier débat public sur l'utilisation des logiciels libres dans le domaine de l'ATM, il a été décidé d'enregistrer tous les exposés et échanges de vues et d'en rendre compte in extenso, cela afin de susciter d'autres initiatives dans l'intérêt du secteur de l'ATM ou de l'OSS. Cette manifestation a d'ailleurs déjà inspiré deux initiatives similaires dans des secteurs autres que l'ATM.

Le présent rapport contient la transcription de tous les exposés et débats qui se sont déroulés à l'occasion de cette table ronde et tente de dresser une synthèse de certaines idées-forces. On trouvera sur le site web www.oss-in-atm.info une version dynamique et sonore du présent compte rendu.

1. INTRODUCTION

par [Jean-Luc Hardy](#), EUROCONTROL

L'ATM fait partie du secteur des logiciels secondaires (*Secondary Software Sector, SSS*), c.-à-d. des domaines d'activité dans lesquels les logiciels sont considérés comme un élément catalyseur ou un composant d'un produit ou service à fournir, mais pas comme le produit final en soi. À ce jour, il n'existe pas de modèle ou d'exemples publiés d'adoption d'un logiciel libre dans le secteur SSS. C'est pourquoi il y a lieu de lancer de nouvelles initiatives en vue de permettre une telle adoption. Le processus qui sous-tend ces initiatives doit être décrit, expliqué et analysé, afin de permettre l'apport d'améliorations, la diffusion des résultats et l'éclosion de concepts de modélisation novateurs.

La table ronde organisée par EUROCONTROL en décembre 2005 constitue un exemple d'initiative prise dans le but de susciter un débat sur l'adoption des logiciels libres dans le domaine de l'ATM. La présente introduction décrit deux caractéristiques propres à la table ronde : le principe de l'échange d'idées et le compte rendu intégral des travaux, destiné à promouvoir d'autres initiatives dans le prolongement des discussions.

1.1. A PRIORI : LE PRINCIPE DE L'ÉCHANGE D'IDÉES

Dans la [brochure](#) rédigée par les organisateurs pour présenter l'objet de la table ronde, on pouvait lire les explications suivantes :

À la mi-2005, le département de la recherche innovante du Centre expérimental d'EUROCONTROL a sollicité le concours du projet CALIBRE en vue d'organiser une table ronde sur le thème suivant : « Possibilités offertes par les logiciels libres dans le domaine de la gestion du trafic aérien (ATM) ».

L'objectif poursuivi par EUROCONTROL est de promouvoir une plus grande sensibilisation ainsi que de recueillir des faits et arguments en rapport avec quatre hypothèses générales concernant les implications possibles du concept de logiciel libre (OSS) pour l'ATM (et plus particulièrement pour le contrôle de la circulation aérienne).

Pour y parvenir, la table ronde a été organisée de manière à promouvoir l'échange d'idées entre deux domaines d'expertise : celui de l'OSS et celui de l'ATM.

Les orateurs étaient invités à se pencher sur les différentes questions qui sont soulevées dans l'introduction de l'ouvrage [Perspectives in Free and Open Source Software](#)⁶

Le programme de la table ronde était structuré de manière à assurer une alternance des exposés sur les initiatives dans le domaine de l'ATM avec les exposés du projet CALIBRE sur l'OSS.

Les initiatives d'EUROCONTROL constituent des actions pionnières sur le plan de la promotion de l'utilisation des logiciels libres dans les projets ATM. Dans les exposés sur l'OSS, l'ATM est considéré comme un cas d'étude possible pour l'introduction des logiciels libres dans le secteur des logiciels dits secondaires (SSS).

1.2. A POSTERIORI : LE COMPTE-RENDU DES TRAVAUX ET LEURS PROLONGEMENTS

Cette table ronde était une première, tant pour les participants au projet CALIBRE que pour EUROCONTROL. Pour les acteurs du projet CALIBRE, il s'agissait du premier atelier organisé à l'intérieur d'une entreprise par le forum industriel CALIBRATION. Pour EUROCONTROL, c'était probablement la première fois que la question de l'utilisation des logiciels libres dans le domaine de l'ATM était étudiée de façon approfondie par des experts issus de ces deux milieux. En effet, ces personnes ont en principe peu de chances de se rencontrer et d'échanger leurs points de vue respectifs : tandis que les spécialistes de l'ATM participent à des projets dans leur domaine de compétence et n'ont pas le temps d'améliorer leur connaissance des logiciels libres en assistant à des conférences sur le sujet, les spécialistes des OSS n'ont guère le temps de s'intéresser à la question spécifique de la mise en œuvre des logiciels libres dans le domaine de l'ATM, malgré le fait qu'ils peuvent immédiatement donner un avis pertinent lorsque ce sujet est évoqué.

La qualité des exposés, le nombre des interactions (216) et la valeur des arguments débattus prouvent que les objectifs a priori qui sous-tendaient l'organisation de la table ronde ont été atteints, tant pour le projet CALIBRE que pour EUROCONTROL.

Pour ce qui est des prolongements de la table ronde pour la communauté ATM, il existe désormais un corpus d'arguments auxquels les acteurs concernés peuvent se référer aussi souvent qu'ils le souhaitent, afin de se familiariser avec le sujet et se forger leur propre opinion sur les possibilités offertes par les logiciels libres dans le domaine de l'ATM. Bien que certains participants s'intéressent dès à présent aux répercussions pratiques de la table ronde sur des projets

⁶ Feller (J.), Fitzgerald (B.), Hissam (S.), Lakhani (K.) [éd.], *Perspectives on Free and Open Source Software*, USA-Cambridge, MIT Press, 2005 (ISBN 0-262-06246-1).

spécifiques, cela prendra du temps et dépendra des décisions prises par le management ainsi que du résultat des négociations avec les partenaires industriels du secteur de l'ATM.

L'ATM est désormais mentionné par les experts en OSS, dans les conférences et ateliers consacrés aux logiciels libres, comme un domaine d'application possible. Une telle étude de cas est bénéfique pour les deux champs d'expertise que sont l'OSS et l'ATM.

En ce qui concerne les suites de la table de ronde pour le secteur des logiciels libres, il est à noter que des initiatives similaires ont vu le jour dans d'autres domaines industriels. Les membres du projet CALIBRE ont ainsi organisé à Madrid, le 4 mai 2006, un second atelier d'échange d'idées chez Vodafone. Un autre est prévu chez PHILIPS, le 19 septembre 2006, à BEST aux Pays-Bas.

La table ronde a également suscité l'intérêt des spécialistes OSS du monde universitaire, qui se sont demandé dans quelle mesure les résultats de cette rencontre pourraient servir de modèle et être reproduits ou transposés dans d'autres contextes industriels. L'utilité de telles rencontres est déterminée par les dispositions prises à la fois avant, pendant et après la tenue de l'événement :

- Le choix des orateurs, une préparation minutieuse du programme de la table ronde et l'agencement de la salle devraient être de nature à stimuler le dialogue entre les experts du secteur ATM, qui présentent leurs projets en faisant part de leur expérience, et les spécialistes des logiciels libres, qui donnent leur avis. La formule de la « table ronde » se caractérise par l'importance accordée aux réactions des participants, aux discussions et à la libre expression des idées, autant de conditions qui sont particulièrement nécessaires au moment de l'émergence d'un phénomène nouveau, comme l'introduction des logiciels libres dans un secteur d'activité déterminé.
- Le rôle premier de l'animateur d'une table ronde est de faire en sorte que le débat demeure axé sur des questions permettant l'enrichissement mutuel des participants.
- Le bénéfice à long terme dépend probablement de la qualité du compte rendu des travaux. L'objectif a posteriori d'une table ronde est d'assurer la diffusion de ces actes. Outre le rapport du CEE, on trouvera sur le site web www.oss-in-atm.info une transcription des débats en format HTML ainsi que les versions électroniques de l'intégralité des exposés et des enregistrements sonores. Le format de ce site web présente quelques caractéristiques structurelles novatrices (non spécifiques au domaine considéré), cela dans le but de souligner l'importance de toutes les interactions résultant du processus d'enrichissement mutuel ¹⁰.

Voici quelques-unes des raisons expliquant le soin tout particulier apporté à la réalisation des actes de la table ronde :

- Le compte-rendu in extenso des travaux devrait souligner l'importance de cette première rencontre pour l'introduction des logiciels libres dans le domaine de l'ATM.
- Les actes devraient contribuer à la diffusion des idées issues du processus d'enrichissement mutuel. Le compte rendu intégral des travaux, et notamment les enregistrements sonores filtrés ¹¹, permettront aux experts qui n'étaient pas présents à la table ronde d'étudier les possibilités offertes par les logiciels libres dans le domaine de

⁷ Accessoirement, deux portions de code Javascript, qui ont pour objet de renforcer la sécurité de ce site web, ont été stockées à l'adresse www.sourceforge.net sous les noms de projet « NoSpamMailto » et « DisableRightClick ».

⁸ Des milliers de bruits parasites et d'hésitations ont été supprimés par filtrage des enregistrements sonores originaux, dont la durée a ainsi été réduite de plus ou moins 20 %. Ces enregistrements filtrés peuvent à présent être écoutés comme si les orateurs lisaient leur exposé d'une manière dynamique. La qualité sonore est néanmoins médiocre pour certaines parties en raison du fait qu'il a parfois fallu travailler à partir d'enregistrements de sauvegarde.

l'ATM comme s'ils avaient assisté en silence à la table ronde. Les hyperliens entre les interventions de chaque participant pourraient inciter les experts à établir entre eux des contacts ultérieurs.

- Pour ceux qui étaient présents à la table ronde, les échanges ont été très nombreux pendant et après la plupart des 15 exposés : le compte-rendu ne recense pas moins de **216** interventions au total. Les sujets abordés lors de cette table ronde étant très variés, il n'est donc pas possible de se souvenir de tous les faits et arguments à l'issue d'une première écoute des enregistrements. C'est pourquoi le site web offre la possibilité de réécouter les différents débats, en naviguant d'une séquence d'enregistrement à l'autre.
- Une seconde chance d'organiser une première table ronde ne se représentera pas. En cas de réédition du même événement, les participants ne seront pas nécessairement animés de la même motivation que celle dont ils ont fait preuve lors de cette rencontre initiale. Des transcriptions détaillées et des enregistrements sonores présentent l'avantage d'exposer les centres d'intérêt, les convictions, les émotions, les craintes, les préoccupations et les idées novatrices des différents participants. Il est vraisemblable que la dimension humaine découlant de ces traits comportementaux sera plus motivante que de simples actes imprimés. Un tel compte rendu in extenso des travaux de cette première rencontre est également essentiel, car il permettra de conserver une trace de toutes les idées initiales et divergences de vues qui se sont fait jour, avant qu'un éventuel consensus ne se dégage.
- Contrairement à d'autres types de manifestations, il ne sera pas facile d'organiser une nouvelle table ronde comme celle-là, en raison du manque de disponibilité des experts qui travaillent à l'intersection des deux domaines considérés. Si une autre table ronde portant sur le même domaine (ATM) devait être organisée, les actes de la première édition serviraient de modèle pour élaborer le programme de la deuxième. Ces actes aideraient à la bonne organisation de la seconde table ronde, tout en évitant les répétitions d'idées et les réunions inutiles. Une analyse de la table ronde pourrait être bénéfique, étant donné que le compte rendu détaillé de ses travaux est censé servir de fondement à d'autres initiatives et décisions portant sur l'introduction des logiciels libres dans le domaine de l'ATM.

2. RÉSUMÉS

2.1. COMPTE-RENDU DES EXPÉRIENCES WWW.OPENATC.ORG ET AUDIOLAN

par [Gilles Gawinowski](#), EUROCONTROL

Résumé. AudioLan et OpenATC représentent plusieurs initiatives en matière de logiciels libres (OSS). Les enseignements tirés de ces expériences tendent à démontrer que les conditions et la motivation pour promouvoir une telle approche ne sont pas réunies :

- La recherche dans le domaine de l'ATM est trop peu importante.
- Le processus de transformation des résultats de la recherche en produits et l'approche axée sur les besoins du client ne sont pas compatibles avec la philosophie OSS.
- Le recentrage sur le métier de base et le mode de gestion de l'externalisation, observés parmi les entreprises, se sont traduits par une diminution considérable des activités de développement logiciel.
- Il n'y a pas d'incitation à surmonter les obstacles à la recherche dans le domaine de l'ATM.

2.2. ÉLABORATION D'UN OUTIL DE PREDICTION DES TRAJECTOIRES SELON LA MÉTHODE DE LA BOÎTE BLANCHE

par [Carlos Garcia Avello](#) et [Sip Swierstra](#), EUROCONTROL

Résumé. De nombreux éléments à la fois essentiels et coûteux du processus de développement du système ATM pourraient être mis en commun, sans pour autant que l'excellence technique et les avantages concurrentiels des entreprises concernées en souffrent. Une approche collective du développement, de la validation et du contrôle de ces éléments communs est la solution la plus efficace pour progresser. Transformer un outil logiciel de prédiction des trajectoires (TP) en une boîte blanche, c'est le rendre aisément accessible à l'ensemble des spécialistes de l'ATM, en même temps que le savoir-faire mis en œuvre et accumulé au cours de la phase de développement. En raison de telles contraintes, l'outil TP ne conviendra pas pour des applications opérationnelles en temps réel. Toutefois, étant donné que le produit sera fourni en tant que logiciel libre, n'importe lequel de ces composants pourrait s'intégrer à des applications existantes. Derrière l'objectif d'offrir aux utilisateurs du TP la possibilité d'analyser le contenu et de décider des choix techniques à opérer, la mise à disposition d'un outil d'une boîte blanche TP a pour but d'ouvrir la voie au développement, sur le modèle des logiciels libres, de composants TP destinés à une large diffusion. Cela permettra d'assurer l'interopérabilité des systèmes clients de prédiction des trajectoires basés au sol ainsi que la synchronisation entre les dispositifs TP embarqués et au sol.

Point le plus important pour l'avenir des logiciels libres dans le domaine de l'ATM. Sans doute l'atout majeur des logiciels libres dans le domaine de l'ATM réside-t-il dans la possibilité d'utiliser ce vecteur pour assurer l'interopérabilité des différents systèmes ATM. La prédiction de trajectoires étant l'une des fonctions essentielles qui sous-tendra le futur concept de gestion du trafic aérien fondé sur les trajectoires, la mise à disposition de composants TP, développées selon la philosophie du logiciel libre, facilitera l'interopérabilité des systèmes ainsi que la compatibilité entre les outils TP embarqués et au sol.

2.3. RECONSIDÉRER LE PROGRAMME CHIPS EN TANT QUE PROJET DE LOGICIEL LIBRE

par [Jean-Dominique Frayssinoux](#), consultant ATM

Résumé. La première partie de cet exposé décrit les objectifs et avantages des logiciels libres dans un environnement ATM. Dans un second temps, l'auteur présente un exemple de logiciel libre, baptisé « CHIPS », qui sera disponible dans un proche avenir. Ce logiciel est un outil conçu pour aider les contrôleurs de la circulation aérienne à résoudre efficacement les conflits de trajectoires entre aéronefs.

Point le plus important pour l'avenir des logiciels libres dans le domaine de l'ATM. Les logiciels libres constituent une excellente occasion de présenter des logiciels utiles et de mettre en valeur l'expertise des spécialistes de l'ATM. La mise à disposition d'un logiciel libre sur un site web permet d'établir des contacts intéressants, dans le monde entier, avec des organismes et des sociétés actifs dans le domaine de l'ATM.

2.4. LOGICIELS COTS VS. FLOSS ET LIBRE JEU DU MARCHÉ DANS LES SECTEURS D'ACTIVITÉ CENTRÉS SUR LA SÉCURITÉ

par [Franco Gasperoni](#), *AdaCore Inc.*

Résumé. Cet exposé analyse, d'un point de vue purement économique, la relation entre les logiciels commerciaux prêts à l'emploi (COTS) et les logiciels libres assortis d'une licence d'utilisation libre (FLOSS). L'auteur s'intéresse en particulier aux secteurs d'activité où la sécurité constitue une priorité essentielle, comme l'industrie aérospatiale. (On trouvera une synthèse complète au début de l'exposé.)

Point le plus important pour l'avenir des logiciels libres dans le domaine de l'ATM. Nos travaux montrent qu'à situation égale, un logiciel commercial bénéficiant d'une licence d'utilisation en toute liberté est toujours préférable, pour le client, à un logiciel commercial vendu avec une licence d'exploitation assortie de restrictions. En effet, l'absence d'un distributeur unique en ce qui concerne les mises à jour, l'assistance technique et la documentation nécessaire pour la certification permet la conciliation des intérêts du distributeur de logiciels COTS avec ceux du client autour de produits et services de haute qualité, et ce à des prix compétitifs. Formulé autrement, sous forme de question : Le secteur de l'ATM devrait-il faire pression sur son (ou ses) fournisseur(s) de façon à ce que, lorsque le jeu en vaut la chandelle, celui-ci (ceux-ci) assortisse(nt) les produits COTS qu'il(s) met(tent) à la disposition de son (leur) client d'une licence d'utilisation de type FLOSS ? Pour l'auteur, la réponse est un oui catégorique.

2.5. LOGICIELS LIBRES ET ÉVOLUTION DU DROIT DE LA PROPRIÉTÉ INTELLECTUELLE À EUROCONTROL

par [Burkhart von Erlach](#), *EUROCONTROL*

Résumé. Le concept de logiciel libre est un phénomène relativement nouveau dans le domaine de l'ATM et à EUROCONTROL. Cette communication traite d'un certain nombre de termes liés à la problématique des logiciels libres. Elle résume la politique actuelle d'EUROCONTROL en la matière, tout en soulignant les raisons pour lesquelles une organisation internationale comme EUROCONTROL se doit d'être prudente, n'ayant pas les coudées aussi franches que le secteur privé. Plusieurs facteurs expliquent pourquoi, jusqu'à présent, les logiciels libres n'ont pas vraiment pénétré le marché de l'ATM. L'auteur mentionne quelques-uns d'entre eux, énumère ensuite certaines des conditions auxquelles les logiciels libres pourraient susciter davantage d'intérêt dans le secteur de l'ATM et, enfin, se demande si EUROCONTROL ne pourrait pas faire de ce champ d'investigation un domaine d'activité à part entière.

2.6. LOGICIELS LIBRES : LICENCES D'UTILISATION PERMISSIVES VS RESTRICTIVES

par [Delphine Prieur](#), *INRIA*

Résumé. La protection juridique des logiciels recouvre deux aspects : les droits patrimoniaux et les droits commerciaux. Une licence est un contrat en vertu duquel ses titulaires se voient concéder des droits d'utilisation plus ou moins étendus. Il s'agit d'un instrument de réglementation. Le concept de licence GPL n'est pas l'antithèse du droit d'auteur. On parle de piratage dès lors qu'il y a violation d'au moins un des droits attachés à un logiciel. Dans cet exposé sont présentés sommairement trois types de licence de premier plan : GPL, CeCILL et la nouvelle licence BSD. Il importe de souligner que le cadre juridique actuel n'a pas encore pu être évalué et qu'il existe des problèmes de compatibilité entre les différents types de licence d'utilisation propres aux logiciels libres.

Point le plus important pour l'avenir des logiciels libres dans le domaine de l'ATM. Lorsque l'on choisit une licence, il ne s'agit pas tant de savoir si cette licence est permissive ou restrictive, mais bien de se poser les bonnes questions quant aux buts poursuivis en matière de développement (qui assurera les développements successifs ? quel sera le statut de ces contributions ?) et de distribution (à qui et pourquoi ?). En plus, il ne faut pas perdre de vue que ces buts peuvent changer au fil du temps, raison pour laquelle il est préférable d'opter pour une licence « à géométrie variable ».

2.7. LA GESTION DES LICENCES D'UTILISATION DES LOGICIELS LIBRES CHEZ PHILIPS

par [Arnoud Engelfriet](#), PHILIPS

Résumé. Voici plusieurs années que la société PHILIPS Electronics utilise des combinaisons de logiciels tantôt libres, tantôt propriétaires, d'où il résulte diverses difficultés à surmonter : comment combiner ces deux types de logiciels en toute sécurité ? Comment gérer les besoins de distribution des logiciels libres ?, etc. L'auteur prend comme étude de cas l'ordonnanceur de disque dur ABISS, développé par PHILIPS (<http://abiss.sourceforge.net/>) et par lequel PHILIPS contribua au noyau du système d'exploitation Linux, afin d'illustrer les meilleures méthodes à mettre en œuvre pour gérer le développement de produits mixtes sur le plan des droits de propriété intellectuelle. Pour différentes raisons, PHILIPS a voulu maintenir le caractère privé de certaines composantes du système ABISS. C'est pourquoi la société a mis au point une méthode d'évaluation permettant de déterminer les composantes qu'il convient de protéger et la manière de procéder.

2.8. TCAS, UN LOGICIEL LIBRE AVANT LA LETTRE

par [Garfield Dean](#), EUROCONTROL

Résumé. Cet exposé établit un parallèle entre le développement des algorithmes du TCAS (système d'avertissement de trafic et de prévention des collisions) et le concept de logiciel libre. Après avoir décrit sommairement le fonctionnement du TCAS, l'auteur fait un bref historique du développement du système. Il explique en particulier les rôles joués dans ce domaine par deux organismes de normalisation : l'Organisation de l'Aviation Civile Internationale (OACI) et la Radio Technical Commission for Aeronautics (RTCA). Pour l'essentiel, l'auteur s'attache à comparer et à mettre en contraste le développement du TCAS et celui des logiciels libres. Enfin, il se demande si, et dans quelle mesure, les logiciels libres seraient utilisables dans le domaine de la gestion du trafic aérien (ATM).

Points les plus importants pour l'avenir des logiciels libres dans le domaine de l'ATM.

- Qui assurera la responsabilité des éventuels développements OSS dans le domaine de l'ATM, notamment en cas d'accident comme celui d'Überlingen ?
- Une autre question connexe est celle de savoir si nous pouvons nous fier aux logiciels libres en ce qui concerne les systèmes vitaux pour la sécurité. Le risque existe de vouloir créer un dispositif qui serait capable de couvrir tous les besoins possibles et imaginables (en l'occurrence garantir la sécurité en toute circonstance), mais qui ne répondrait finalement à aucun (parce que sa complexité de fonctionnement le rendrait inutilisable).

2.9. AMÉLIORATION DE LA QUALITÉ ET GESTION DES MISES À JOUR

par [Martin Michlmayr](#), Université de Cambridge

Résumé. D'aucuns affirment parfois que les logiciels libres et les projets de développement correspondants sont généralement de meilleure qualité que les logiciels propriétaires. S'il est vrai que plusieurs éléments probants donnent à penser qu'une évaluation sérieuse par les pairs contribue à la qualité d'un produit, il n'en reste pas moins un grand nombre de questions restent à examiner. L'exécution des projets de développement de logiciels libres se fait de manière décentralisée, généralement par des bénévoles, ce qui entraîne des difficultés particulières à résoudre. Après avoir donné un aperçu des problèmes de qualité les plus couramment rencontrés, dans les grands comme dans les petits projets de développement de logiciels libres, cet exposé examine plus particulièrement la gestion des mises à jour, une problématique propre à de nombreux logiciels libres.

2.10. LES LOGICIELS LIBRES DANS LE SECTEUR DES LOGICIELS SECONDAIRES : LE POINT DE VUE DU SECTEUR PRIVÉ

par [John O' Flaherty](#), CALIBRE

Résumé. Cet exposé aborde le phénomène du logiciel libre du point de vue du secteur privé et met en évidence un certain nombre de réalités complexes inhérentes au rôle joué par les logiciels libres dans le secteur des logiciels secondaires (*Secondary Software Sector*, SSS). Il présente les résultats d'une recherche menée dans le cadre d'un atelier international, qui a été organisé dans le but clairement affirmé de faire entendre la voix des principaux partenaires industriels. Les informations ainsi recueillies ont été analysées selon une méthode qualitative, qui a mis en évidence les atouts et faiblesses des logiciels libres du point de vue du secteur privé. Cette démarche a jeté les bases de l'élaboration d'un cadre de référence décrivant les nouveaux logiciels libres qui commencent à se répandre dans le commerce (qui sont qualifiés d'« Open Source Software Inc. »). L'auteur conclut son exposé en affirmant que les acteurs du marché européen des logiciels secondaires reconnaissent certes les avantages qu'il y a à exploiter le potentiel des logiciels libres, mais sont conscients des problèmes majeurs qui subsistent en la matière.

3. BILAN PRÉLIMINAIRE : LES LEÇONS À TIRER

par [Marc Bourgois](#), EUROCONTROL, 15 décembre 2005

Des échanges de vues stimulants et parfois très animés qui ont eu lieu, nous pouvons tirer les cinq enseignements suivants :

- Toutes les entreprises sont désormais confrontées au phénomène des logiciels libres.
- Le problème des licences d'utilisation n'est pas un obstacle insurmontable.
- Le succès d'une communauté de développeurs dépend de certaines conditions.
- Les logiciels libres permettent de créer de la valeur ajoutée.
- Les domaines d'activité vitaux pour la sécurité sont totalement délaissés par les projets pilotes en matière de logiciels libres.

3.1. TOUTES LES ENTREPRISES SONT DÉSORMAIS CONFRONTÉES AU PHÉNOMÈNE DES LOGICIELS LIBRES

La première leçon que nous avons tirée, probablement la plus fondamentale, s'est fait jour après que nous nous sommes posé la question « À quoi bon ? ». Pourquoi l'Organisation EUROCONTROL devrait-elle en effet se préoccuper des logiciels libres ? La réponse est à la fois très simple et très générale, en ce sens qu'elle vaut pour toutes les organisations qui jouent un rôle actif dans le développement d'applications logicielles : toute organisation ou entreprise est confrontée au phénomène des logiciels libres, qu'elle le veuille ou non. Le risque est grand qu'une application acquise dans le commerce inclue certains modules dont le code source est librement accessible.

Toute application mise sur le marché est susceptible d'être intégrée ultérieurement dans un logiciel libre. Et il ne s'agit plus seulement des fournisseurs ou des clients, puisque même les développeurs (bien connus pour leur indépendance et leur liberté d'esprit) sont aujourd'hui enclins à s'en remettre à certains logiciels libres.

Selon le type de licence d'utilisation qui accompagne un logiciel libre, les risques encourus peuvent aller de pair avec des conséquences commerciales importantes. Ainsi, dans un cas, la société PHILIPS a dû distribuer des dizaines de milliers de CDs, tandis que, dans un autre, elle n'a pas eu d'autre choix que de mettre son produit sur le marché en tant que logiciel libre par suite de l'inclusion, par inadvertance, de composantes de type OSS.

Comment maîtriser les risques inhérents à l'inclusion, par mégarde, de modules à code source libre dans un logiciel ? Trois solutions sont possibles : (i) analyser le logiciel au moyen d'outils spécialisés en vue de détecter d'éventuels éléments de code source libres, puis éliminer la dépendance par rapport à ceux-ci ; (ii) déterminer, sur la base d'une stratégie, les logiciels pour lesquels le libre accès au code source présenterait un intérêt commercial, et ensuite concevoir une architecture permettant de dissocier les composantes logicielles propriétaires des parties de code source libres ; (iii) définir une politique interne (qui devrait inclure les deux mesures précédentes) visant à sensibiliser davantage le personnel de l'entreprise à cette problématique.

Au cours des discussions de la table ronde, il est apparu très clairement qu'il règne une grande confusion parmi les spécialistes de l'ATM (et sans doute parmi ceux de n'importe quel autre domaine confronté, depuis peu de temps seulement, au phénomène des logiciels libres) quant à savoir quelles applications méritent vraiment le statut de logiciel libre. Nous avons déjà constaté ce problème dans le cadre de nos travaux sur le terrain, dès avant la tenue de la table ronde. Qui plus est, le concept de logiciel gratuit (*freeware*) est souvent assimilé, à tort, à la notion de logiciel libre (OSS), entre autres parce que l'un comme l'autre vont souvent de pair avec des communautés d'utilisateurs et de développeurs très dynamiques. À cet égard, on ne citera jamais assez cette analogie célèbre que l'on doit à Richard Stallman, l'un des maîtres à penser du concept de logiciel libre : *free as in free speech, not as in free beer* (en français, « libre au sens de liberté, et non de gratuité »). Le monde du logiciel libre est lui-même victime, depuis ses débuts, de cette ambiguïté terminologique. Il n'est dès lors pas étonnant, compte tenu de la multitude d'expressions en usage (*OSS, Libre Software, Free Software, FLOSS*), que les juristes présents à cette table ronde aient commencé, dans leur exposé, par définir le concept de logiciel libre, ce qui nous amène au deuxième enseignement tiré de nos discussions.

3.2. LE PROBLÈME DES LICENCES D'UTILISATION N'EST PAS UN OBSTACLE INSURMONTABLE

Une séance de brainstorming, animée à l'occasion d'une récente conférence réunissant les membres du projet CALIBRE, a permis de classer par ordre de priorité les problèmes auxquels il convient de remédier pour que le concept de logiciel libre recueille une large adhésion. Les questions juridiques figuraient à la sixième et dernière place de ce classement. La concession des licences et les autres questions connexes ne devraient manifestement pas mobiliser les foules. Cette opinion a d'ailleurs été corroborée lors de notre tour de table.

Il existe un nombre effarant de licences différentes régissant l'utilisation des logiciels libres (environ 200). Cette situation présente à la fois des inconvénients et des avantages : elle est néfaste, parce qu'elle crée la confusion dans les esprits et fait fuir les non-juristes ; mais elle a aussi du bon, dans la mesure où la licence d'utilisation qui répondrait aux besoins spécifiques de votre projet existe probablement déjà et qu'il ne vous reste plus qu'à l'adopter.

S'il est vrai que les questions de propriété intellectuelle qui entourent les logiciels libres sont complexes (même les juristes ne s'accordent pas nécessairement sur tous les points, comme nous avons pu le constater lors de la table ronde), il n'empêche que les multiples licences d'utilisation existantes peuvent être regroupées trois grandes catégories : (i) les licences « *Free-for-All open source* » ; (ii) les licences « *Keep-Open open source* » et (iii) les licences « *Share-Alike open source* », qui prévoient le partage à l'identique des conditions initiales et s'inspirent du marketing viral pour assurer la diffusion du produit. Ces catégories se fondent sur les deux obligations fondamentales incombant au titulaire d'une licence OSS : la mention du code source d'origine et/ou le partage avec autrui des modifications et des éventuels ajouts effectués.

En fait, l'Agence EUROCONTROL a déjà développé une application qui relève de la deuxième catégorie des logiciels OSS. Il s'agit d'un utilitaire rédigé en ADA, qui a été réintégré dans la version OSS de base, à la demande des utilisateurs externes. S'il est vrai que ce module logiciel ne peut être considéré comme une application ATM à proprement parler, cette expérience n'en a pas moins fourni une première occasion, pour le Service juridique de l'Agence, de se familiariser avec ces questions, et pour le management, de prendre position sur le sujet.

Aussi la solution au problème de la licence consiste-t-elle à faire preuve de proactivité en spécifiant avec précision les besoins stratégiques / commerciaux du projet, ce qui exige de faire intervenir les juristes à un stade précoce afin de réduire les risques et de déterminer la solution la plus appropriée.

3.3. LE SUCCÈS D'UNE COMMUNAUTÉ DE DÉVELOPPEURS DÉPEND DE CERTAINES CONDITIONS

L'atout le plus fréquemment cité des projets OSS est sans nul doute leur communauté de développeurs, (que l'on dit) très nombreux et prompts à réagir. Les partisans du concept de logiciel libre attribuent la qualité et le succès durable des projets OSS à l'abondance et à la gratuité des contributions d'une équipe de développeurs bénévoles largement disséminée.

Ainsi qu'il a été montré, maintes de ces affirmations ne se vérifient que pour un très petit nombre de projets OSS. Se prononcer sur la qualité des logiciels libres n'est pas chose aisée, et le verdict n'est pas invariablement le même. Les chercheurs qui s'intéressent aux logiciels libres ont à peine commencé à approfondir l'étude de la relation éventuelle entre les particularités de la communauté OSS et la qualité des logiciels qui en sont le fruit. Certains éléments portent à croire que l'ensemble des outils de développement utilisés ainsi que la motivation des développeurs sont des facteurs qui pourraient être en corrélation avec la qualité des logiciels libres.

On observe actuellement un nombre sans cesse croissant de développeurs qui apportent leur contribution à des projets OSS pendant leur horaire de travail normal, et ce avec l'accord sans réserve de leur employeur. Il faut voir dans cette tendance le fait que le monde du logiciel libre parvient à la maturité dans certains secteurs du développement logiciel, principalement ceux liés à l'infrastructure ou ceux qu'il est convenu d'appeler les domaines horizontaux. De toute évidence, cette évolution vient battre en brèche l'affirmation initiale selon laquelle un grand nombre de développeurs apportant leur contribution bénévolement est une condition sine qua non de la réussite d'un projet OSS. Aujourd'hui, l'analyse se veut plus subtile, en considérant que la qualité des contributions (qui, par définition, est relativement élevée parmi des professionnels) est plus déterminante que le nombre des collaborateurs. Cela est vrai dans un domaine vertical pointu comme l'ATM, qui aura toujours du mal à attirer la communauté, nettement plus nombreuse, des développeurs « amateurs », qui travaillent à domicile durant leur temps libre. Les participants à la table ronde ont néanmoins quelque peu réfléchi à la manière dont un domaine aussi hermétique que celui de l'ATM pourrait mobiliser et motiver la communauté des développeurs de logiciels libres dans son ensemble. Parmi les idées qui ont été formulées, l'une des plus prometteuses a trait au développement d'un simulateur ATM (à l'instar des simulateurs de vol).

Comme il a été dit plus haut, plusieurs spécialistes de l'ATM ont présenté aux participants à notre table ronde les résultats préliminaires d'expériences menées en matière de logiciel libre. Il est intéressant de noter que deux de ces projets OSS ont été qualifiés d'« échec ». L'un concernait un portail / forum créé par EUROCONTROL, en coopération avec un autre grand centre de recherche actif dans le domaine de l'ATM. L'objet de ce portail était d'héberger des logiciels libres, mis à la disposition de l'ensemble de la communauté des chercheurs ATM, en vue de leur utilisation, de leur analyse critique et de leur évolution. Étonnamment, bien que plusieurs outils de recherche aient été téléchargés, le forum n'a suscité que peu de discussions, voire aucune. Cette situation est demeurée inexplicable, jusqu'à ce que les spécialistes OSS nous éclairent sur l'existence de deux facteurs, difficilement quantifiables, qui se révèlent déterminants pour le succès d'une communauté d'adeptes des logiciels libres : la motivation et le militantisme.

D'une part, il apparaît essentiel d'associer très tôt cette communauté, afin d'éviter une probable réaction de « rejet du corps étranger » lorsque ses membres se sentent mis devant le fait accompli. Au lieu de contributions toutes faites, il serait préférable que la relation entre l'organisation contributrice et la communauté OSS soit fondée sur la règle d'or suivante : *release early, release often* (« diffuser rapidement, diffuser souvent »).

D'autre part, toute initiative nouvelle doit être menée par un défenseur énergique, un « dictateur bienveillant », capable de piloter l'évolution souvent chaotique d'une communauté de développeurs qui trouvent leur motivation en eux-mêmes. Ces deux enseignements méritent d'être pris en considération sérieusement lors de la sélection de projets pilotes portant sur l'adoption de logiciels libres.

3.4. LES LOGICIELS LIBRES PERMETTENT DE CRÉER DE LA VALEUR AJOUTÉE

Revenons à présent au deuxième exemple de projet de logiciel libre dans le domaine de l'ATM, qui a été qualifié d'« échec ». Il s'agit d'une application baptisée AudioLan, qui, à l'origine, a été développée pour le compte d'EUROCONTROL et qui a pour but de remplacer les commutateurs téléphoniques de type analogique par la technique de la transmission vocale via Internet (VoIP) pour les communications sol-sol entre contrôleurs. (Reste à voir si la demande de mise en œuvre d'un tel système pour les communications air-sol peut être justifiée.) La version préliminaire de ce logiciel a été rapidement adoptée dans plus de 10 centres ATM européens, représentant plus de 500 postes de travail de contrôleurs.

Pour la seconde fois, les spécialistes OSS présents à la table ronde ont avancé une explication plausible. Se fondant sur leur vaste expérience, qui leur permet de faire des parallèles, ils ont soutenu avec force que ce phénomène s'observe chaque fois que le module de logiciel libre est l'élément déterminant pour différencier les produits des acteurs industriels en compétition. Pour qu'un logiciel libre soit adopté avec succès par des compétiteurs industriels traditionnels, il faut que l'application en question ne constitue pas l'élément central de l'avantage concurrentiel de leurs produits respectifs.

Dans une telle situation, des concurrents acharnés peuvent, assez paradoxalement, se comporter de manière altruiste, en contribuant en grande partie à la normalisation de la composante OSS. Un bon exemple de ce phénomène, qui provient d'un domaine horizontal, est l'implémentation de X-Windows : cette application a connu un bel essor grâce au soutien financier d'un consortium industriel, parce qu'elle était considérée, non pas comme un élément contribuant de façon décisive à la différenciation du produit, mais bien comme un élément de nature à faciliter l'interopérabilité exigée par les utilisateurs. X-Windows est peut-être l'exemple le plus connu d'implémentation de référence, qui est devenue un cas d'exception, mais elle n'est absolument pas le seul exemple en la matière. Nous pensons que cet enseignement est de bon augure quant à l'adoption réussie des logiciels libres dans le domaine de l'ATM.

Comme il a été rappelé dans l'interview avec le représentant du projet CALIBRE, qui a précédé la table ronde, la question a été posée de savoir si les fournisseurs du secteur ATM ne constituent pas déjà une industrie de services, et ils se leurreraient eux-mêmes s'ils faisaient comme si l'essentiel de leurs bénéfices provenait d'applications ATM de base, et non de l'intégration et de la maintenance des systèmes ainsi que d'autres activités de service.

3.5. LES DOMAINES D'ACTIVITÉ VITAUX POUR LA SÉCURITÉ SONT TOTALEMENT DÉLAISSÉS PAR LES PROJETS PILOTES EN MATIÈRE DE LOGICIELS LIBRES

Enfin, pour ce qui est du cinquième et dernier enseignement – sur lequel il n'y a pas lieu de s'étendre –, nous avons pu constater l'absence totale de pénétration des logiciels libres sur le marché des applications vitales pour la sécurité. Aucun exemple n'a pu être trouvé dans la littérature spécialisée ni cité par le groupe des experts OSS présents à la table ronde. Il convient cependant de noter l'existence de nombreuses observations empiriques du dynamisme des communautés OSS, qui sont très promptes à réagir à des relevés d'erreurs graves de programmation, et ce avec bien plus d'empressement que ce qui serait normalement spécifié dans des contrats conclus avec des fournisseurs d'applications propriétaires vitales pour la sécurité.

Est-ce à dire que le secteur de l'ATM ne peut pas bénéficier des possibilités offertes par les logiciels libres ? Certes non. Cela signifie simplement que les études en la matière ne doivent pas porter en premier sur les applications ATM vitales pour la sécurité. Le système ATM dans son ensemble comporte de nombreuses composantes non essentielles pour la sécurité, qui sont autant de possibilités d'accumuler de l'expérience dans le domaine des logiciels libres. De fait, un spécialiste de l'ATM présent à la table ronde a soutenu, avec compétence mais aussi avec un sens certain de la provocation, que les applications ATM ne sont nullement essentielles pour la sécurité du fait que les conflits de trajectoires entre aéronefs sont, par essence, résolus en permanence, les opérateurs humains disposant toujours de quelques minutes de réaction pour faire face aux éventuelles pannes des systèmes automatisés. Et l'intervenant de poursuivre en affirmant que les seuls véritables composants vitaux pour la sécurité sont les équipements électroniques de bord. Compte tenu des développements futurs du système ATM, qui ne peuvent aller que dans le sens d'une plus forte intégration air-sol, il est certain que la frontière entre les applications qualifiées de vitales pour la sécurité et celles qui ne le sont pas deviendra plus floue que jamais.