# How to benefit from Open Source components in your embedded products?

Pierre Gaufillet, Clotilde Marchal

# How to benefit from Open Source components in your embedded products?

Pierre Gaufillet pierre.gaufillet@airbus.com, AIRBUS
Clotilde Marchal clotilde.marchal@eads.net, EADS
With the participation of other members of the open source working group of EADS

Keywords: open source, business model, governance, intellectual property, export control

Summary

Integrating open source components in embedded products is a real opportunity to improve your productivity and to implement attractive new features in your products. But for benefiting safely from this new approach of Intellectual Property, a good knowledge of its characteristics and a minimum of organization are needed. For two years now, an EADS working group has been exploring and implementing such organizations in each business unit. Based on this experience, we will go in this article through the risks induced by the availability of open source components, and will expose an approach for implementing the control mechanisms that are required in your company.

---

The last 10 years have seen the emergence of open source in most industries. Software tools, Telecommunication, IT and more recently aerospace, energy, automotive and healthcare have at least made a step toward open source. It is not a fashion effect: due to the features of open source licenses, open source code is accumulating innovations and value with the time. It makes more and more difficult for companies not using it to remain productive enough compared to their competitors leveraging open source. In other terms, open source has become a valuable source of productivity. Recent surveys have also shown that even if you think you are not using open source components, there are nevertheless probably some of them in your applications and in some of your embedded products – even if safety/mission critical softwares are obviously better controlled due to their strict development and verification process. Beyond the fact that some major pieces of code – like Linux, Apache or Eclipse Rich Client Platform – are open source, it is due to the advent of generalized high bandwidth Internet accesses combined to efficient search engines and a cultural change of developers. Combine that with more and more complex supply chains and you will soon realize that it is difficult to know where the source code of your application is coming from.

In itself, it may remain technically manageable, as far as you ensure that all parts in your code are integrated and verified in a consistent way. But there are several families of open source licenses, and some of them can have unexpected legal effects on your Intellectual Property. Copyleft licenses for example, also known as reciprocal licenses, cover not only the open source components themselves, but also the original code linked to them, potentially extending to the whole application. Integrating code under copyleft licenses may therefore result in having an obligation to give access to your customers to the whole source code of your application. Of course, it is most of the time possible to solve that either by modifying the architecture of the application, or by changing your license to open source – and therefore probably change your business case, which is not so obvious. Beyond the management of the Intellectual Property, integrating open source components – and more generally 3rd party components – implies to take care of other important aspects:

- Export control: if your applications are to be delivered to customers in a list of foreign countries, you may face regulation issues, most often related to the exportation of cryptographic algorithms. It is quite common to integrate components that include such algorithms even if you are not using them. In practice, you may even ignore their presence. PDF generators are a good example, as most of them provide RC4 encryption capabilities. Depending on the kind of algorithm exported, you may be required either to declare it, or to ask for an export authorization to the regulator.

- Security: depending on the architecture of the application, 3rd party components may introduce vulnerabilities. Of course, it can be easily managed at design time, first by checking the list of known vulnerabilities, coming for example from the NIST database. But it doesn't solve everything. You will also have to set up an organization capable of analyzing the impacts of newly discovered vulnerabilities during the whole life time of your product. And when required, you will need the capability to integrate security patches, rebuild and distribute new releases of your application.

- Professional support: one of the most common arguments heard against open source in the industry is the lack of professional support. Of course, this vision is the result of the decentralized nature of open source. It means that in most case it is not one but several companies that can provide support – including on demand warranties and liabilities. Open Source foundations may also play a role here, ensuring that some fundamental features are maintained, like IP control and technical hosting of their projects.

- Certification: depending on the criticality of your applications, integrating 3rd components can be more or less difficult, and can be addressed thanks to several strategies ranging from a full technical appropriation of the source code – a kind of corporate fork – to the integration as a black box.

For these reasons, after preliminary studies made in several business units, EADS headquarters have launched in 2011 a working group aiming at implementing the organization required to benefit safely of open source components in its tools and products. During almost 2 years, this working group has explored most of these topics and it has quickly become clear from the list of risks identified above that properly using – and not using – open source requires some organization and means. This is now part of software development as other activities like specification, design or functional verification. This experience has shown that there are several ways to implement it, depending on the size of the company and its business models. EADS business units are now in a pilot deployment phase aiming at refining the governance and verifying that the required tools can be integrated smoothly in the corporate IT infrastructure.

But the first and most important step is certainly to create a group of experts gathering IP legal counsel, export control officer, procurement, open source specialists and software architects. The correct analysis of the risks and the resulting decisions require indeed numerous skills and can be seen as a shared responsibility. Of course, SMEs may not have all the dedicated resources for this purpose, and big companies may consider these activities as outside of their core competencies. It is still possible in this situation to rely on external services, which can be provided by specialized actors like law firms and open source service experts. Whatever organization has been chosen, the second step is to train all the employees potentially facing open source including software developers, project managers, architects and system designers.

Organization, process and trainings are of course fundamental, but it also makes sense to support at least a part of the activities that have been already identified by specific tools:

- Open Source portfolio management tool: in numerous cases, embedded products have to be maintained for quite a long period. It is therefore important to log a number of information for each product: what open source components are being used, how they are integrates, what version is used, etc. Thanks to this information, it is not only possible to take the right decision years later, but also to automate some expansive activities like security follow-up. Managing an open source component obviously implies costs, so maintaining this portfolio is also a way to reduce future analysis – by replying on past decisions for example – and to limit the number of components in use – by reminding architects during future selection process that a component with similar features has already been deployed.

- Code scanner: a number of very efficient code scanners are available on the market. While not absolutely required (en fait je dirais que c'est plutôt obligatoire non ? ou alors c'est un risque à prendre...), these tools are a very good mean to raise the level of confidence you have in the IP analysis of your application, especially in the context of complex supply chains. It is to be noticed that these code scanners can operate of course on source code, but also on binary resources like dynamic libraries, images and other multimedia resources. Some of them also provide a formal analysis of the compatibility of licenses. Several integrations are possible, from a simple verification on build, to early verification during design review, through continuous verification of each code commit in your version control system.

Of course, if both tools are deployed, integrating those makes sense and gives the ability to improve their efficiency. The code scanner can use open source portfolio information to improve its detection capabilities, and the portfolio can incorporate the results of the scan, giving management an overview of open source usage for all the projects of the company. It is also possible to generate documentation, for example a booklet gathering all the licenses included in the product.

To summarize, whether or not you want to use open source components, protecting the Intellectual Property of your company in the current situation requires introducing at least a few verifications in your acquisition

and development process.