TAMPERE UNIVERSITY OF TECHNOLOGY

# Legal issues in ICT product development projects – shortly and practically

Matti Vuori

24.9.2014

# Contents 1/2

# Contents 2/2

# Introduction

- This is a short and practical look into the legal issues that a project group could face.
- The idea is to help the group identify issues and associated risks and to handle them appropriately in all phases of the project.
- So, in a sense this is a checklist in a form of a slide set.
- This is obviously incomplete – just a starting point to raise awareness.

# General challenges

- The whole ICT industry is only gradually learning these things.
- Accidents happen with IPRs and contracts causing problems for business later.
  - Someone else claims they have a copyright, patent, exclusive license or something to something that we wish to use.
- We need to learn to be more proactive and sharper with all this.

# Good news!

- Legal things have never been an issue on the project course.
- So, don't be too scared.
- …But still you need to be aware of these matters during the course and later in your career.

# Basis for everything: simple agreements between customer and group

- Before any legalities, the group and the customer must agree on who owns the results.
  - (Nowadays, in many cases, open sourcing is a good solution.)
- In our course, oral agreement has been used in majority of the cases.
- But you need to know all obstacles, so we take a look into the issues and then how they can be addressed in projects.
-

# License 1/3

- You can license a product to others in various ways:
  - Transfer all rights – then you can do nothing.
  - Give rights to use, sell, modify – and retain the same for you.
  - Give a right only to use – most common.
  - Release as open source under suitable OSS license – and let anyone do with it what they want. (This makes often plenty of sense.)
  - Etc…

# License 2/3

- If you retain all rights to the product, you can sell licenses to many parties.

# License 3/3

- For 3rd party components in your product, licenses are critical:
  - Are you allowed to use something at all?
  - Can it be distributed as part of your solution?
  - Does an open source license stick to your code and make it open source too?

- So: assess all licenses very carefully!

# Contract 1/5

- Customer and client need a contract between them do define (among others):
  - What the group promises to do.
  - Under which terms.
  - Who owns the results – who can and can not use the results.
  - Confidentiality – what can be told about the project outside.
  - Liability – in case of problems in the market.

# Contract 2/5

- There are many contract types:
  - Project, maintenance, hosting, non-disclosure agreement (NDA), subcontracting, license…
- But the contract is just one of the papers that bind you.
- Everything that is agreed by parties count:
  - Project plan.
  - Maintenance plan.
  - Specifications.
  - Etc…

# Contract 3/5

- Contract is most valuable when there are problems or when there is great success.
  - Product sold to Facebook for 10000000 euros…
- Oral contract is valid, but write it down because memory is fallible and people change in ICT organisations often.

# Contract 4/5

- Contract is a business issue first, after that legal.
  - Business = what both parties want to do.
- Think of it as professional practice and courtesy towards the customer.

# Contract 5/5

- Use a good template as basis, but industrial templates are not good in this context.
- Keep things simple and clear! Make sure you understand and double check everything. Make sure the other party understands things exactly the same way…
- Long legal documents lead to imbalanced situation, since companies are experts.
- Members of TEK can use their free legal counselling.
- Many domains use general contract such as IT 2010 (http://www.it2010.fi/) and Finnish government JIT 2007 /JHS167 (http://www.jhs-suositukset.fi/suomi/jhs166)
- Note: Supplier (project team) has the IPRs of things they produce unless otherwise agreed.

# Good news!

- In the project course, the customers are always friendly.
- They wish good things for you and don't want to rip you off.
- So making contracts with them is usually a pleasure.

# Confidentiality 1/2

- Customers may require NDA. Preferably at company level.
  - Defines what you may and may not talk about the project to outsiders.
- Project group should be able to use project as reference, tell about it to others.
- It is just professional integrity not to talk about customer's business.

# Confidentiality 2/2

- Customers may require:
  - No printing in public places (TUT student printers).
  - No use of USB sticks – they are easy to lose.
  - Encryption of emails and attachments.
- Large companies have more such rules.
- Need to check these at the start of project.

# Potential contract risks

- Is the work covered by contract or billed as extra? So specify clearly.
- All rights given to customer, cannot self develop product further.
- Is the product implemented as agreed?
- Bound by NDA, project cannot be used as a reference.

# IPR protection

- The elements of a product (ideas, design, implementation…) may have various types of protection.
- This will help you
  - To protect your work.
  - To respect others' rights by not violating them.
  - To not get caught in legal trouble.

# Basic product idea

- Alternatives:
  - No protection – every product much like others…
  - Some central idea may be patented.
- Myth: Patents mostly apply to hardware
  - Software really is patented.
  - When doing something unique, this is worth checking.

# Sources of patent info

- Google Patents (patents.google.com) has excellent coverage of US patents (the main area of software patents), but also European patents.
  - Good because of its linking of patent data to citing of patents, rich displays of patent data and other benefits.
- Espacenet (http://fi.espacenet.com/quickSearch?locale=fi_fi ) contains patent information from 80 countries.

# Some realism…

- Patent searches are overkill for student projects.
- They take time and require some skills in reading the "patent language" where often information is hidden in purpose with abstract wordings.
- But remember this when thinking of commercialising something unique, especially in USA.

# Product name 1/2

- Names are protected either by being registered trademarks or simply by being already in use (this depends on the country).
- Note that same name can usually be used in another domain (food vs. sports), but better avoid duplicates – not just for legal reasons, but to lessen confusion when googling…

# Product name 2/2

- Final product names etc. should be checked carefully.
- But a product can have a work name during development (companies often have silly code names for products).
- Don't hard code names anywhere… names will change.

# Visual design, industrial design

- Visual design can be copyrighted or has a formal design right (see http://www.prh.fi/en/mallioikeudet.html) obtained by registration – such as a cover design for a device.
- Common problem in software development:
  - Design is contracted, but customer has not bought rights to modify designs. Has to buy any modifications from designer.

# 3rd party code


- You may NOT use just any piece of code found in the internet.
- Binary code is also copyrighted (obviously).
- Code is copyrighted even if published.
- **So check the license carefully**.
- **If in doubt, don't use**. Rewrite the implementation yourself.

# Example 1/3

- Suitable code is found in WWW page or discussion forum:
  - Try to find the original development repository, where the code comes from.
  - Check license info found in code and in project documentation (often LICENCE file if repository root).
  - If there is no such info, don't use the code.
  - Analyse licensing for whether you can use the code or not.
    - Find out requirements for documenting and displaying that your program contains this code under some license.

# Example 2/3

- If you can use the code at all, you may need to:
  - Mark your derived source files  to show what it is derived from.
  - Retain the original copyright and licensing info in your code.
  - Include the license file of that code in your project's user and developer documentation.
  - Show in your product's online information (such as an About screen) that your product contains this code under this license.
  - In case of license conflicts, need to reconsider your licensing.

# Example 3/3


- ABSOLUTE DO NOT just change variable names and indenting to make code "yours". That will not work.

# Own code

- Basic principle: the party who wrote the code, owns copyright to it.
- Employee rights in Finland:
  - Copyright to most other things is owned by individual employee, but software is an exception in copyright law! The employer owns it.
- Mark every code file with copyright information in the header.

# Tackling all this during project

- Next we'll take a look at how to tackle all this during a project.

# Project planning 1/2

- During risk analysis, identify risks related to IPRs and how to deal with them.
- Make sure that all team understands these issues.
- Make contracts carefully.
  - Discuss things with customer openly, they are not evil
  - Be careful with inexperienced customers.
  - Don't agree on impossible terms.

# Project planning 2/2

- Check all product, technology names that you can really use them.
  - This in critical only when you publish them.
- Get external advice if uncertain about anything.
  - You can always ask course personnel for opinions, advice.

# Choosing open source licenses 1/2

- If your product is open source based, plan the license carefully, considering:
  - Your future plans.
  - Licenses of any main components that you plan to use.
- Analyse the risks of going open source carefully.

# Choosing open source licenses 2/2

- About open source licenses:

http://opensource.org/licenses/category

- Comparison of licenses:

http://en.wikipedia.org/wiki/Comparison_of_free_and_open-source_software_licenses

# Concept design

- Check if the product ideas or key solutions are protected by patents or by other means.
- This is often the customer's task – but someone must do it.

# Requirement specification

- Check for legal or standard based requirements of the target markets / nations / domains.
  - For example, safety critical domains may require you to use certain design practices (and development practices) and you need to know those beforehand.
  - The customer *should* know these.

# Visual design

- If you are subcontracting design work (GUI, logos, templates), see that you have the right to modify designs later.

# Feature development

- In design reviews look for potential infringements – but any serious research is overkill.
- Review code and check if you have the right to use all of it – check origin of all code.
- **When choosing components, check the licenses** – be careful about using open source components – the licence issues.

# Delivery, publishing, productisation 1/2

- Check that your work fulfils all contractual requirements.
  - Contracts between parties
  - License terms, including documenting what components have been used under which license
- Double check for compliance with laws and standards in the target markets / nations / domains.

# Delivery, publishing, productisation 2/2

- Check that you don't violate any trademarks, copyrights etc…
  - Product name first to check.
- Have a plan for pulling to product back if there are problems.

# Document things 1/2

- Document all issues, assessments, discussions, decisions related to laws, IPRs, etc.
- Including discussions with customer.
  - Make memos of meetings.
  - Write notes of phone call conversations.
  - Save emails.

# Document things 2/2

- Document all code and product artefacts: who made them, when.
  - For example source code file header information.
- Never delete any documentation or artefacts.
  - Disc space is cheap nowadays.
- Retain version control database if possible.

# Appendix: Some abbreviations

- NDA: Non-disclosure agreement
- OSS: Open Source Software.
- FOSS: Free Open Source Software
- FLOSS: Free/Libre/Open Source Software