Chapter 7

Warranties and Liability

Agile ideology emphasizes "working software over comprehensive documentation" (Beck et al., 2001). This can be misinterpreted so that in agile development no documentation outside the source code is needed. Having no documentation than the source code will cause serious problems at least when defining warranties for the system, thus some kind of lightweight documentation is needed also in agile software development projects.

This chapter takes a look to the warranties and liabilities in traditional Finnish contracting law and practice. Next section points out what additional difficulties iterative development, where features can be accepted in separate phases, can cause to defining warranties. Then the warranties and liability clauses in general contracting terms are investigated, and finally the case contract's warranty terms are explained.

7.1 Warranties in traditional contracting

Based on the Finnish Sale of Goods Act §32, the reclamation period starts when the buyer should have noticed the fault in the product. It is also possible to define in the contract that the supplier will guarantee that the product will have certain features, which decreases the need to notice the faults, or define a period when the customer needs to complain about the found faults (Hemmo, 2005).

In practice, in traditional software development, the finished software is delivered to the customer who then tests it during the acceptance phase. During the acceptance tests the customer will report all bugs found from the system, usually refusing to accept the delivery before the program is perfect. This can make the project last forever, thus the contracts often include a term that "good enough" (e.g. ready for production use) software has to be

accepted and found faults are then fixed without additional cost during the warranty period. (Takki, 2002)

After the complaint the supplier has per se the right to correct the faults on their own expense if it does not cause significant harm to the customer. If the corrections have to be ordered from someone else, the costs can be charged from the supplier, and if the faults are significant and cannot be fixed, the customer has the right to terminate the contract. (Hemmo, 2003) Because of the nature of software programs, the contract can also include a clause that the supplier will do their best during the warranty period to fix all the bugs found during the acceptance tests as well as on the warranty period, but will not guarantee that the program will be completely bug-free. (Takki, 2002)

If there are major faults in the program after the warranty period, the customer can claim a rebate. The customer can also withhold part of the payment, e.g. 15 % of the price of the product, to be payed only after the errors are fixed during the warranty period. This will work as supplier incentive to fix the errors in order to get the customer to pay the last consideration. That is not usually mandatory since if the supplier acts like a scoundrel what comes to fixing the errors in the software, it will probably not stay in the business for long. (Takki, 2002)

The contractual division between the work done for the customer and the produced product that Takki (2002) discusses also affects the liabilities. Commitment to the result (produced product) requires that the exact, concrete outcome is accomplished, where the commitment to the work done requires that the work is performed in properly and carefully but the outcome is not thoroughly defined. If the supplier does not accomplish the outcome when the contract is about the produced product, the customer has the right to claim for a new accomplishment, fixing the error(s) or reduced price. On the other hand, if the contract is about the work done, previously listed consequences are only possible if the supplier has not been performing properly. (Hemmo, 2003)

Traditionally, the negligence (or lack of it) is not a factor when evaluating the insufficient fulfillment of the profit responsibility: it is not possible to avoid the responsibility by pleading to the fact that one has tried to fulfill their responsibilities. Instead, supervisory responsibility is used, meaning that the supplier can be released from the responsibility only if the contract breach is caused by an obstacle that they cannot impact or bypass and which they could not have known on the moment of contracting. (Hemmo, 2003)

In traditional software development projects the warranty for the system is usually defined so that the finished product matches the functional specification. The fact that the warranty does not mention the actual expectations

of the customer is meant to rule out the Sale of Goods Act §17 responsibility that the product must be suitable for the purpose those kind of products are usually used as well as the customer's special use of which the supplier has been aware. For example, the warranty period may exclude the right for discount. Using the warranty as limitation to liability is not legal in consumer business but usual in contracting between companies: in about 99 % of the cases the customer would be entitled to larger compensations without the warranty clauses. (Takki, 2002)

7.2 Agile warranties

Singe in agile projects comprehensive functional specification is not usually developed, this makes defining warranties trickier in agile projects. To help defining the scope of warranties in agile projects, each delivery should contain summarized user stories of the features included in the delivery. Together these stories will describe the overall functionality of the system and effectively work as a functional specification of the system. (Opelt et al., 2013) This document should, naturally, be developed in cooperation with the customer so that both parties will have the same understanding of the user stories and how they fulfill the product vision. In addition to being useful, low-effort overview of the product, this product description can also be used as a basis of the supplier warranty.

Iterative development also means that features are finished in different times which adds an extra challenge for defining the start of the warranty period. There is a possibility to start the warranty period of each deliverable after the iteration when the feature is accepted, use single starting point for the whole system's warranty or use combination of the two methods. Despite the chosen warranty specification, it should be certain that the supplier accepts the responsibility for the whole project. (Opelt et al., 2013) Takki (2002) notes that if not otherwise specified, no partial acceptance will relieve the supplier from the responsibility that the outcome of the project must correspond to what was originally agreed.

If combination of the two warranty styles is used, at the end of each iteration warranty is tied to the the incremental working deliverable. In addition, there is an overall warranty to the whole product, which starts after the final acceptance. (Arbogast et al., 2010) However, this can lead to complex situations if, for example, sprints 1-8 are accepted (and their warranty period has started), sprint 9 is rejected (thus no warranty) and then sprint 10 is again accepted. This would mean that warranty periods run for parts of the current system but not for the whole system. (Opelt

et al., 2013)

The simplest way of defining warranty in agile projects is, in fact, to define it similarly to traditional project models: a single start date of the warranty is right after the final acceptance of the project. This acceptance is separate from the acceptance of the final iteration and is performed against the product description which is developed during the iterations. (Opelt et al., 2013)

7.3 Warranties in the general contracting terms

In IT2010 general terms the warranty of the software is defined in the EJT special terms. After the supplier has tested the software and delivered it to the customer, the customer has 30 days to perform acceptance tests for it and notify the supplier about all errors found. If the system has an error that prevents the testing, the tests can be suspended until the error is corrected and the testing time is extended by the length of the delay caused by the corrections. (IT2010 EJT, 8.3) Errors that are not severe (i.e. the system can be taken or is taken into production use) will not prevent the acceptance of the system but the supplier does have to correct the errors without delay (IT2010 EJT, 8.4). The supplier will then correct all the reported errors at no cost during the warranty period, which lasts 6 months from the acceptance of the delivery (IT2010 EJT, 11.1). If the system is delivered in phases, the limitations above do not apply to the acceptance of a partial delivery if it has an error that could not have reasonably been noticed before to the acceptance tests of a later partial delivery (IT2010 EJT, 8.5).

JIT 2007 has a similar acceptance phase than IT2010. After the software is delivered to the customer, they have 30 days to perform an acceptance inspection to it and report all found errors in the software. If the software is delivered in phases, the customer will inspect each delivery separately within 7 days of the delivery, and the next phase cannot be started before the previous one is accepted. In an agile project, this would mean unbearable break between each short iteration. The acceptance of interim phases do not discharge the supplier from liability for errors found later in the development or in the final acceptance tests. Minor errors should not prevent the acceptance of the software but the supplier is required to fix those errors at their own cost and without unreasonable delay. Also, if the customer starts the production use of the software, the product is seen as accepted. (JIT 2007, 10)

JIT 2007 states that the ”product and end result of the service must be free of errors”. All the errors have to be fixed by the supplier at his own

expense without a delay, and the supplier has to pay a contractual penalty for the time used for removing the errors after the agreed delivery time. However, this penalty does not apply to situation where the error is a minor one or if the supplier repairs it immediately. If the error prevents the usage of the product, the customer can also withhold the payment of the part of the product that is affected by the error until the supplier fixes the problem. (JIT 2007, 16)

The warranty period of the software in JIT 2007 terms is defined separately in the Applications annex. It is 12 months from the date of the customer's acceptance of the application, and if the application is accepted in phases, the warranty of each part will not expire until 6 months have elapsed from the acceptance of the whole application. (JIT 2007, Applications 6.1) If the product is unusable during this period due to an error, the warranty period is exceeded by that period of time (but not beyond twice the original warranty period). If the supplier does not correct the errors in a reasonable time, the customer can order the repairs from a third party and charge the expenses from the supplier. (JIT 2007, 17)

The customer is entitled to compensation for direct damages (and not for indirect damages) caused by delay or other breach of the contract to the extent by which the damage exceeds the contractual penalty. If the breach is not due to negligence, the compensation should not be more than 7.5 % of the price of the product, but if the damage is due to negligence, the maximum compensation is as high as the total price of the product. Limitations on liability do not apply if the damage is caused deliberately or through gross negligence. (JIT 2007, 21)

Warranty for the system is defined in the Swedish general terms so that if the software has errors caused by the supplier not being skilled enough in their work, the supplier should, at his own expense, correct the errors. These errors should be highlighted by the customer no later than six months after the termination of the project (of three months after the error should have been discovered if it have been discoverable earlier). If the supplier is unable to fix the errors in a reasonable time, customer is entitled to compensation. (General Terms for Agile Projects, 9.1) Unless presence of intent or gross negligence, the amount of compensation is limited to 20 % of the total amount paid for the project, if not otherwise agreed (General Terms for Agile Projects, 10.2).

As said, in the Swedish contract terms, the liability for defects is related to the supplier's workmanship in performing their work. If the supplier's work is professional and skilled, errors found during the iterative development are handled similarly than any other requirements: they continuously change and are prioritized accordingly. In addition to that, the supplier is liable for

errors caused by regression i.e. errors that are caused by the new features not working together with the old ones. The fact that supplier is liable only for damages occurred due to their negligence in performing their work (General Terms for Agile Projects, 10.1) is a difference to traditional contracting where the lack of negligence does not redeem the supplier from the compensations, but in practice it might be hard for the supplier to prove that their work has been professional and skilled.

The Norwegian PS2000 terms take a more traditional approach to warranties: the supplier is responsible for the work done just like in traditional contracting. The terms are similar to the ones in IT2010 and JIT 2007 terms. After the supplier delivers the product, the customer will perform acceptance tests for it, and the project is ended in a common evaluation meeting where both parties can present their possible financial claims towards the other. (Laine et al., 2011)

7.4 Warranties in the case company

The case contract defines warranty period to be 6 months after the go live of the system, and it assures that the work done corresponds to what has been defined. Even though the software is developed and delivered in short iterations, the acceptance of the partial deliveries do not affect the supplier's overall responsibility of the product. The whole product is accepted after the whole system has been delivered and acceptance tests for it have been performed.

During the warranty period, the supplier will fix all the errors found from the system at their own cost. After the warranty period, the supplier is still obligated to fix at their own cost all critical errors found; critical error meaning a error that prevents using the software or parts of it or makes the usability of the software significantly worse. If the supplier does not correct the errors in a reasonable time or if similar errors are continuous despite the supplier's actions, the customer can order the repairs from a third party and charge the expenses from the supplier.

The customer is also entitled to compensation for direct damages (and not for indirect damages) caused by delay or other breach of the contract to the extent by which the damage exceeds the contractual penalty. If the breach is not due to negligence, the compensation should not be more than 7.5 % of the price of the product, but if the damage is due to negligence, the maximum compensation is as high as the total price of the product. Limitations on liability do not apply if the damage is caused deliberately or through gross negligence.