Ethical Responsibility of the Software Engineer

Gonzalo Génova[(1)], M. Rosario González[(2)], Anabel Fraga[(1)]

[(1)] Departamento de Informática, Universidad Carlos III de Madrid
Avda. Universidad 30, 28911 Leganés (Madrid), Spain
{ggenova, afraga}@inf.uc3m.es

[(2)] Departamento de Didáctica y Teoría de la Educación, Universidad Autónoma de Madrid
Ciudad Universitaria de Cantoblanco, Cra. Colmenar Viejo, km. 15, 28049 Madrid, Spain
charo.gonzalez@uam.es

Abstract. Among the various contemporary schools of moral thinking, consequence-based ethics, as opposed to rule-based, seems to have a good acceptance among professionals such as software engineers. But naïve consequentialism is intellectually too weak to serve as a practical guide in the profession. Besides, the complexity of software systems makes it very hard to know in advance the consequences that will derive from professional activities in the production of software. Therefore, following the spirit of well-known codes of ethics such as the ACM/IEEE's, we advocate for a more solid position, which we call "moderate deontologism", that takes into account both rules and consequences to assess the goodness of actions, and at the same time pays an adequate consideration to the absolute values of human dignity.

1. Introduction

The moral progress of society is highly influenced by the way we reason in the various fields of ethics, and in particular professional ethics. The laws that govern a society are responsible for the structure it acquires in the long term. Yet it is the task of ethical thinking to inspire the development of laws. Each one of us implicitly acknowledges the primacy of ethics over law when we cry out: *this law is unjust!* (Think of laws about racial discrimination, minimum salaries, and so on.) Apart from the brute force (of weapons, or of votes), the only other force that can change the laws is the ethical reason.

This is why ethical thinking is so important in everyone's education: if our moral arguments are weak, we are at the mercy of the best speaker. In particular, it is crucial in the education of modern professionals, such as software engineers, because the ethical thinking is not only made up of abstract principles, but it is also derived from the real professional life and circumstances. If you want to formulate ethical judgments about rates of interest, taxes and salaries, you must be knowledgeable about this notions in the field of economy. In the same way, to judge about the moral responsibility of the software engineer requires a good knowledge of the profession, well aware of the experience and the real way engineers work.

Every engineer is first of all a free person, an ethical agent. Ethics, far from being a set of limits imposed on our freedom, is the precise way we become our own masters. Without a specific and solid ethical education, the engineer becomes a mere technical, depersonalized instrument in the hands of others.

This has been recognized in many places and educational institutions. In particular, the *Computing Curricula* developed by ACM/IEEE, which is taken as an exemplar for many university programs, puts a significant emphasis to ethics and law courses in Chapter 10, devoted to Professional Practice, and promotes various strategies for incorporating them into the computer science curriculum [2].

In this paper we are not concerned with general ethical issues in Information Technologies, such as privacy of personal data, freedom and censorship in the Internet, intellectual property of software products, intrusions, frauds and abuses committed with the aid of, or against, software systems, and so on. We rather want to focus on ethical issues that more directly concern the responsibility of the software engineer in the production of faulty software systems, and the bad consequences that can be derived from them. Software systems are powerful systems which can cause severe harms to human lives or well-being, and when this occurs we want to know who is responsible, who shall pay for it. But this analysis must not ignore that it is in the very nature of Software Engineering to deal with the production of complex systems, where the consequences of actions are particularly difficult to predict.

This paper is structured as follows. Section 2 presents the notion of responsibility. Section 3 surveys the distinction between rule-based and consequence-based ethics, and how these two approaches entail a different notion of responsibility. Then, Section 4 applies these notions to the problem of complexity in the production of software systems. Finally, Section 5 contains a summary of our argument and some concluding remarks.

2. The notion of responsibility

The term "responsibility" has a variety of senses [8]. We can distinguish between *role responsibility* and *causal responsibility*. A person is playing a "role of responsibility" when she[1] has some duties or obligations because of her function or position in society. For example, parents are responsible for their children, they cannot abandon them. The reality surrounding someone *demands an adequate response* from that person; the possibility of not responding is excluded: not to act is one way to react [12]. The adequate response involves, first, a *clarification* of the situation to discover the values at stake, and the exact measure they demand a response from the agent; and second, a *prioritization* of the potential courses of action, since our limited nature impedes us to satisfy all possible demands. All this requires open-mindedness and dialogue with reality.

On the other side, we talk of "causal responsibility" when we look for the sources of certain results or consequences in the *actions or omissions* of an agent. Since the

[1] To avoid the continuous repetition of "he or she", in this paper we will use "she" to denote the generic third person.

effects have usually a multitude of causal factors, in practice we are trying to identify the abnormal factor in an unexpected effect. For example, if a forest is burnt, we will consider as normal factors the capacity of wood to burn, and the presence of oxygen in the atmosphere; but the facts that someone lit a bonfire (action) and the firemen did not react (omission) will be regarded as the abnormal factors that caused the forest to be destroyed. The notion of *blameworthiness* or *culpability* can be associated with role responsibility, but more often with causal responsibility: when a person is responsible in this sense, we expect from her to *repair* the bad consequences of her actions or omissions: for example, the consequences of a faulty program code.

Another common distinction is found between *accountability* (i.e. moral responsibility) and *liability* (i.e. legal responsibility). There are situations where the law will require some kind of repairing or compensation to the harms caused (strict legal responsibility), even though there was not properly a bad action from the moral point of view: for example, if there is a failure with more or less severe consequences, despite the software was honestly produced with all reasonable efforts to assure its quality and following the highest standards, then the software company will be liable.

Nevertheless, *moral responsibility is generally broader than legal responsibility*. As we have seen, ethics inspires the development of law, but one of the functions of the law is to put clear limits to responsibility in social life, so that it can be prosecuted with the instruments of power, such as penalties, etc. If the laws demanded from us all that ethics does, our lives would become unbearably regulated. Real life is richer than the laws can reflect, and excessive laws can even suffocate our freedom to do it better than it is strictly demanded by law. Besides, ethics pursues an *internalization of values* that acquaints oneself with good, and eases to capture the demands of the situation and to give an adequate response to those demands[2]. But this internalization is out of the scope of law, which is satisfied with an external submission. In summary, the ethical behavior cannot be confined within a code of conduct.

3. Rules vs. consequences: is there a clear boundary?

Contemporary schools of ethics can be organized in very different ways. A very common distinction among them is that of "rules vs. consequences" [9]. Ethicists who are in the "rules" camp believe *good actions result from following the correct rules* of behavior, which generally are thought to be universal and applicable to all; the rules must be followed regardless of the consequences, good or bad, that might result. Ethicists who focus on consequences, in contrast, believe general rules are not specific enough to guide action and feel instead that we must look to the consequences of our actions, and *take the actions that produce the best results* or consequences. Technically, this distinction is known in the ethics literature as "deontologism" vs. "consequentialism"[3]. In a famous 1919 lecture, the sociologist

[2] This is what ancient philosophers like Aristotle and Seneca called "virtue".

[3] A common distinction is made between action-based and rule-based consequentialism [8]. They respectively consider the consequences of individual actions, or the long-term consequences of applying general rules. This distinction does not affect the core of our argument, and therefore we will not deal with it for the sake of brevity.

Max Weber, who contributed to the general acceptance of this distinction, gave them the names "ethics of conviction" (*Gesinnungsethik*) vs. "ethics of responsibility" (*Verantwortungsethik*) [13]. The first position has more an air of *honorability*, whilst the second one seems more *flexible and reasonable*: they could represent the hero we admire and the pragmatist we follow (using the words of Weber, the saint and the politician).

We will call these two extreme positions "rules-without-consequences" and "consequences-without-rules" (see Figure 1). What many do not perceive is that these two positions *cannot resist the slightest rational analysis*, therefore they do not truly represent realistic ethical positions that are worth considering as practical guides for action.

Extreme Deontologism	Moderate Deontologism	Moderate Consequentialism	Extreme Consequentialism
Rules without Consequences	Rules with Consequences	Consequences with Rules	Consequences without Rules

Is there a clear boundary?

Figure 1. A panorama of contemporary schools of ethics

Let's take first the *rules-without-consequences* ethical position. There is no rule of behavior which ignores at all the consequences of the actions, since it is completely impossible to define an action without considering its precise effects: acting means producing effects [11]. The rules "thou shallt not lie", or "thou shallt not murder" are not inconsiderate to consequences: they are precisely forbidding very concrete consequences, i.e. lies and murders. In other words, extreme deontologism, if it really tries to disregard consequences, cannot propose practical rules.

On the other side, the *consequences-without-rules* ethical position is irrational for different reasons. First, the consequences of a certain action extend over a period of time that properly has no limit, yet we cannot indefinitely wait to judge whether an action is good or bad. Second, even if we put a timely boundary to the consequences we want to consider, they nevertheless belong to the time to come, therefore they are uncertain; we should employ some kind of prediction technique to foresee the consequences and valuate them; but these techniques will always be limited by the very nature of things, which do not follow perfectly known behavior rules (besides, consequences will probably depend on the freedom of others). Third, and most important, if we want to avoid *a priori* rules of goodness for actions, and we make the goodness of an action depend on the goodness of its consequences, then we need rules to valuate the goodness of the consequences[4]; extreme consequentialism does not solve the problem of goodness, but simply puts it off. In summary, "take the actions that produce the best results or consequences" does not designate anything practical.

[4] This reveals also that consequentialism is not "value-neutral": it requires a set of values or rules, as well as deontologism does. Neither deontologism nor consequentialism can be ethically neutral, and of course they should not be. There will be a variety of deontologist and consequentialist ethical systems, depending on the set of values they choose to respect.

Summing up, both extreme positions are neither rational nor practical (they are not only unreasonable, but also impossible to put into practice), so that the contrast between them is more misleading than helpful to understand ethical concerns [11]. In fact, too often deontologism or consequentialism are presented in their extreme, irrational forms, because it is easier to mock at them if you are contending from the opposite camp.

Well then, what can we say about the rationality and practicality of moderate positions? Let's call the two moderate positions as "rules-with-consequences" and "consequences-with-rules" (see Figure 1). Is there a smooth transition from one side to the other, or is it sharp, on the contrary?

The moderate deontologist, who stands in the *rules-with-consequences* ethical position, knows she cannot take refuge in rules of behavior that are to be followed blindly. Instead, she has to consider the consequences of the actions for a proper valuation of ethical responsibility: severe consequences will weigh more than minor ones, and short-term consequences will be considered before long-term ones. The consideration of the consequences will influence the way to apply the rules, so that *at least some of the rules will not be absolute*. Of course, a good reason (possibly derived from the consequences) is required to depart from these rules that are not absolute: "relative" does not mean one can follow the rule at free will.

Conversely, the moderate consequentialist, following the *consequences-with-rules* ethical position, knows that calculating best outcomes requires some *a priori* rules to put a boundary to the consequences to be considered, and to valuate the goodness of those consequences. Apparently, this leads to a position which is very similar to the moderate deontologism. So, where is the difference, if any?

Here. The true consequentialist, even if moderate, *will not acknowledge absolute rules* that forbid some kinds of actions: *everything enters into a calculation*, so that an action that is considered bad under certain circumstances will become good in a different situation. On the contrary, the moderate deontologist does not believe every rule is unconditional, yet *at least some of them are absolute*: some actions are considered "bad in themselves", they are never ethically permitted, even though they could have some indirect good consequences.

In other words, there is definitely a clear boundary between moderate deontologism and moderate consequentialism, which is the acknowledgment of certain *unquestionable barriers* not to be trespassed at all. This principle has been traditionally stated as "the (good) end never justifies the (bad) means", and it is derived from the *recognition of human dignity*: thou shallt never do something that directly harms human dignity. Human dignity cannot enter into calculations.

This is not a purely theoretical principle: it has practical implications. The difference between moderate deontologism and moderate consequentialism is not purely abstract. Therefore, we cordially disagree with Don Gotterbarn [5]: the different ethical theories are not simply different conceptualizations of essentially the same way of acting.

Now then, *what is human dignity*, and what are those unconditional rules? Well, the answer is not easy, and it is out of the scope of this work. Different cultures have different notions of it, but the idea of human dignity is nevertheless a conquer of civilization, explicitly acknowledged as such in the constitution of our modern society [14, Preamble]. Briefly, we could say that human dignity has to do with the

uniqueness of every human being, which is valuable in itself. Even if we know it imperfectly, pursuing human dignity, that is, the constant effort and compromise to know and protect it better, is a fundamental ethical option that makes a difference[5].

4. Responsibility in the production of complex software systems

Software systems are strongly characterized by their complexity: both the production process and the delivered product are very complex realities, even more if we take into account the great demand of more distributed, networked and heterogeneous systems [10]. We can consider two different implications of complexity with regard to ethical responsibility. First, responsibility will be *shared* among a large set of people, from the requirements engineer to the software architect, from the designer to the tester, and above all the project manager. Who will be responsible for failures in the delivered software, for going beyond the budget, for missing deadlines in the schedule? What if people jobs, houses or lives depend on the software system working properly, or simply being delivered on time? This demands *a clear distribution of responsibility* among the different roles in the production process.

Second, complexity makes consequences much more unpredictable, and therefore *a consequentialist analysis of responsibility* becomes much harder. Everybody would be very happy if we had a perfect way to predict the behavior of a software system, but software systems are not a mechanical world of ideal billiard-balls. Unfortunately, computer science is not at a stage where we can have, in general, formal proofs for the proper functioning of a software system or subsystem. Most of times, systems are not formally proven (because we don't even know how to); they are simply tested. If we had to wait for a perfect demonstration that the software has no bugs, then no software would be delivered ever, and this is not what society expects from software engineers. Tests can be (and should be) methodical and rigorous, and many software engineering standards contain specific guidelines and procedures to achieve good testing of software[6]. But we cannot expect from tests the mathematical demonstration that the system will have no bugs. Software Engineering, like Medicine, is not Mathematics.

The point here is that *this imperfection and unpredictability of software belongs to the very nature of the software engineering profession*, as we know it nowadays, and as it is publicly and socially recognized. In fact, we find similar imperfection and unpredictability in other fields of engineering. The civil engineer can, and must, adopt reasonable measures to avoid a bridge break-down, but she cannot mathematically

[5] A last remark. When we say "absolute" or "unconditional" rules we do not mean "rules that are perfectly known and fixed for ever", but rules that, as far as we know and accept them, forbid a behavior (bad in itself) that cannot be compensated by other indirect good effects or consequences through some kind of balance or calculation. As we have already stated, our notion of human dignity improves (has improved and hopefully will improve) over history, therefore our absolute rules to protect it have to evolve equally.

[6] See for example the European Space Agency Software Engineering Standards, *Guide to Software Verification and Validation* [4]. Note that this standard explicitly acknowledges that formal proofs are exceptional.

assure it. If she adopts those reasonable measures, well known in the profession, then she will not be responsible for the collapse, if it ever happens.

Besides, we need a way to put boundaries to the consequences that the developing team should be responsible for. Clearly, being responsible for "all consequences" of one's acts would be too much for anyone, in any profession, especially if consequences are difficult to predict. For example, the team could be regarded as responsible for delays in the project delivery, but it would be unreasonable to blame them for failures in the old system that had to be replaced with the new one. On the other side, predicting the social consequences of the use of software systems is generally out of the competence of software engineers, which are not (and don't need to be) trained in social sciences [3]. The engineer, as any other one in her life and profession, needs to know that her moral and legal responsibility is limited: since she has a limited capacity of action, she must have a limited responsibility. Otherwise, the risk to suffer abuses in arbitrary assignment of responsibilities is too big.

Summing up, *a purely consequentialist analysis of responsibility reveals inadequate in the field of software engineering*: first, because consequences are extremely difficult to predict due to the complexity of software systems; second, because no human being can be held responsible for *all* possible consequences of her actions (think of the "butterfly effect"). Of course, we do not pretend this to be an excuse to ignore at all the consequences of one's acts: our intention is to show that extreme consequentialism (consequences-without-rules), besides its inherent lack of rational consistency, is particularly inadequate in software engineering. Since extreme deontologism (rules-without-consequences) is equally irrational and impractical, we must choose between one of the moderate positions, rules-with-consequences or consequences-with-rules.

Both of them contain the implicit notion of *foreseeable consequences*, those consequences that may be reasonably foreseen and for which the agent will be held responsible. Note that this notion imposes *a precise obligation on the agent to foresee*, not to disregard, the consequences of her acts. Note also that we talk about the foreseeable consequences, not the *effective results*, which could be different from predicted, therefore not being the agent accountable (morally responsible) for them. Again, this is not an excuse for a bad prediction, but it acknowledges the weaknesses of prediction methods. Even though a professional could be liable (legally responsible) for the effective results, she could be exempted of moral responsibility if they were honestly not predicted, and legal penalties should be softened in that case.

Moreover, even if bad consequences are correctly foreseen and they become effective results, in some cases the agent should not be held accountable or liable. For example, if general purpose software such as a word processor or a mail system is used to plan and commit a robbery. Therefore, only *direct consequences* of the actions (or omissions) should be taken into account to assess moral and legal responsibility. Here we can distinguish between *software failures*, which in general entail moral or legal responsibility, and *software misuses*, which do not. But, again, we need to consider some nuances about misusing software. If a software system is intentionally designed to spy on banking transactions and commit a fraud, then the software designer is clearly responsible for the software misuse, even though she does not use it directly: it is to be held a direct consequence of the software design. On the other side, if someone unintentionally provokes a strong harm because of a user interface

misinterpretation in circumstances of emergency, then maybe the interface designer should not be completely exempted of responsibility: but this situation is not so clearly a direct consequence of a bad design.

The point that we want to emphasize is that assessing the direct consequences that can be reasonably foreseen requires *a good knowledge of the profession*: someone outside of the profession cannot do a good prediction, and therefore is not properly qualified to make ethical judgments about the matter. However, these two qualifications of consequences (*direct* and *foreseeable*) are vague and open in the end. Ethical principles cannot be used as the input to an ethical algorithm that generates ethical decisions: the agent cannot avoid her personal ethical judgment in each particular situation [1, Preamble].

In spite of their weaknesses [3], current codes of ethics in software engineering provide valuable guidelines to achieve ethical behavior and to assess moral responsibility in the profession. At first sight, they appear to be more rule-oriented than consequence-oriented; but, in fact, they are full of rules about the way to consider consequences. In other words, they adopt what we have called a *moderate deontologist* ethical position: a view where *human values are preeminent* (including some absolute values whose violation cannot be compensated by "good consequences") and, precisely because of that, *the consideration of the consequences of actions is crucial*. We cannot minimize the importance of "computing consequences" in advance to assess moral responsibility and to choose the most adequate course of action [7], but this would become *a meaningless calculus* if we forget that the ultimate goal is preserving a due respect for human dignity. *Human values are the framework that provide sense to the assessment of consequences.*

Let's take a closer look at the ACM/IEEE code: "The Code helps to define those actions that are ethically improper to request of a software engineer or teams of software engineers" [1, Preamble]. In a medical code of ethics we can expect to find values centered about the due respect to human life, because caring for human life is the goal of Medicine. Not surprisingly, since software engineering deals with *information*, the main values that we find in the clauses of this code are those that concern *truth*: fairness and sincerity[7], honesty and integrity[8], earnestness[9], etc. Other clauses are more relative and empirical in nature, since they state convenient ways to avoid bad practices that, according to the experience of many years of profession, would lead to defective processes and products: appropriate testing, good documentation, and so on. Above all, the "concern for the health, safety and welfare of the public is primary; that is, the *public interest* is central" [1, Preamble]. A centrality that cannot be achieved through the contrast and opposition of rule vs. consequence ethics, but rather with a good integration of absolute values, empirical rules, and pragmatic consideration of the consequences of actions.

---

[7] Be fair and avoid deception in all statements, particularly public ones, concerning software or related documents, methods and tools [1.06].

[8] Provide service in their areas of competence, being honest and forthright about any limitations of their experience and education [2.01].

[9] Be accurate in stating the characteristics of software on which they work, avoiding not only false claims but also claims that might reasonably be supposed to be speculative, vacuous, deceptive, misleading, or doubtful [6.07].

5. Conclusions

We have examined and contrasted two different schools of ethical thinking, deontologism vs. consequentialism, and we have shown our preference for the first one, in accordance with the spirit of current codes of ethics in software engineering. The following lines summarize an outline of the argument:

1. **Extreme positions are neither rational nor practical:**
   - 1.1. Deontologism cannot ignore the consequences in defining prescribed or forbidden actions.
   - 1.2. Consequentialism cannot ignore the rules in deciding which consequences are relevant and in assessing the goodness or badness of consequences.
   - 1.3. The contrast between extreme positions is not helpful but misleading: opposing rules to consequences is a false problem.
2. **Moderate positions are both rational and practical, but not equivalent:**
   - 2.1. Both integrate rules and consequences to ethically valuate actions.
   - 2.2. Moderate deontologism acknowledges some absolute or unconditional rules, whilst moderate consequentialism does not.
   - 2.3. Absolute behavior rules in moderate deontologism are derived from the recognition of human dignity, which cannot enter into calculations.
3. **Software systems are strongly characterized by their complexity:**
   - 3.1. Complexity and imperfection of software makes the prediction of consequences particularly difficult.
   - 3.2. This imperfection and unpredictability of software belongs to the very nature of the profession, as we know it nowadays.
   - 3.3. This is not an excuse to ignore at all the consequences of one's acts, but shows that a consequentialist analysis of responsibility becomes much harder and inadequate in software engineering.
4. **We need a way to put limits to the consequences for which a professional will be held responsible:**
   - 4.1. Focus on direct and reasonably foreseeable consequences.
   - 4.2. Assessing the direct consequences that can be reasonably foreseen requires a good knowledge of the profession: someone outside of the profession cannot do a good prediction, and therefore is not properly qualified to make ethical judgments about the matter.
   - 4.3. Ethical principles cannot be used as the input to an ethical algorithm that generates ethical decisions: the agent cannot avoid her personal ethical judgment in each particular situation [1, Preamble].
5. **Current codes of ethics in software engineering provide valuable guidelines:**
   - 5.1. They adopt a moderate deontologist ethical position.
   - 5.2. They are aware of the problems derived from software complexity and do not try to teach precise (algorithmic) mechanisms to valuate responsibility.
   - 5.3. They strive for a good integration of rules and consequences to achieve ethical behavior and to assess moral responsibility in the profession: preeminence of human values and crucial consideration of consequences.

Good intentions are not enough. Software engineers require a sound ethical instruction that integrates moral principles and respect to human dignity with the real experience of their profession (what we have called *moderate deontologism*). A code that is "founded in the software engineer's humanity, in special care owed to people affected by the work of software engineers, and in the unique elements of the practice of software engineering" [1, Preamble] is especially adequate to "educate and inspire software engineers" [6] to achieve ethical behavior, and to learn that they are free persons with a responsibility that cannot be transferred to others.

The task of the moral philosopher is not to indicate a closed, perfectly defined path. Rather, she has to teach how to discover, how to interpret and how to understand the signs found on the way, to discern good from evil, and to invent new ways to do good.

References

1. ACM/IEEE. *Software Engineering Code of Ethics and Professional Practice*, v5.2, 1999. Available at (www.acm.org/serving/se/code.htm).
2. ACM/IEEE. *Computing Curricula 2001, Computer Science, Final Report*. The Joint Task Force on Computing Curricula, IEEE Computer Society and Association for Computing Machinery. December 15, 2001.
3. Frank Bott, Allison Coleman, Jack Eaton, Diane Rowland. *Professional Issues in Software Engineering*, 3rd Ed. Taylor & Francis, 2001.
4. European Space Agency. *ESA PSS-05-10 Guide to Software Verification and Validation.* ESA Board for Software Standardisation and Control, March 1995.
5. Donald Gotterbarn. "The Moral Responsibility of Software Developers: Three Levels of Professional Software Engineering". *The Journal of Information Ethics*, 4(1): 54-64, 1995.
6. Donald Gotterbarn, Keith Miller, Simon Rogerson. "Software Engineering Code of Ethics is Approved". *Communications of the ACM*, 42(10):102-107, October 1999.
7. Chuck Huff, C. Dianne Martin. "Computing Consequences: A Framework for Teaching Ethical Computing". *Communications of the ACM*, 38(12):75-84, December 1995.
8. Debora G. Johnson. *Computer Ethics*, 2nd Ed. Prentice Hall, 1994.
9. Kenneth C. Laudon. "Ethical Concepts and Information Technology". *Communications of the ACM*, 38(12):33-39, December 1995.
10. Ian Sommerville. *Software Engineering*, 7th Ed. Pearson-Addison Wesley, 2004.
11. Robert Spaemann. *Moralische Grundbegriffe*. München, 1982.
12. Paul Watzlawick, Janet Beavin Bavelas, Don D. Jackson. *Pragmatics of Human Communication*. W. W. Norton & Company, 1967.
13. Max Weber. "Politik als Beruf". *Gesammelte Politische Schriften*. Mohr, Tübingen, 1958.
14. United Nations. *Universal Declaration of Human Rights*. December 10th, 1948. Available at (www.un.org/Overview/rights.html).