Prioritizing Legal Requirements

Aaron K. Massey[†], Paul N. Otto[†,‡], and Annie I. Antón[†]
Department of Computer Science, North Carolina State University[†]
School of Law, Duke University[‡]
{akmassey, pnotto, aianton}@ncsu.edu

Abstract

Requirements prioritization is used in the early phases of software development to determine the order in which requirements should be implemented. Requirements are not all equally important to the final software system because time constraints, expense, and design can each raise the urgency of implementing some requirements before others. Laws and regulations can make requirements prioritization particularly challenging due to the high costs of non-compliance and the substantial amount of domain knowledge needed to make prioritization decisions. In the context of legal requirements, implementation order ideally should be influenced by the laws and regulations governing a given software system. In this paper, we present a prioritization technique for legal requirements. We apply our technique on a set of 63 functional requirements for an open-source electronic health records system that must comply with the U.S. Health Insurance Portability and Accountability Act.

1. Introduction

In his 2004 State of the Union address, President George W. Bush set the goal of providing every American citizen the ability to have Electronic Health Records (EHRs) by 2014 [7]. In February 2009, President Barack H. Obama signed into law the American Recovery and Reinvestment Act,[1] which reserves $17 billion exclusively for the development of EHR systems [22]. President Obama's goal is the same as President Bush's goal: to provide EHRs to Americans by 2014 [14]. Such high-level bi-partisan agreement marks the development of EHRs as a clear national priority for the United States.

In the United States, the Health Insurance Portability and Accountability Act (HIPAA)[2] regulates EHR systems, such as those promised by Presidents Bush and Obama. The HIPAA established safeguards to protect the privacy and security of health information. Penalties for non-compliance with HIPAA are severe: for a non-criminal violation of the HIPAA Privacy Rule, violators could be fined up to $25,000 per year per violation.

Serious penalties for non-compliance raise the question of where software engineers should begin. Requirements prioritization is the process of determining the order in which requirements should be implemented. Prioritization has been an active area of study in requirements engineering for decades [1, 8, 10, 11, 12, 13, 20, 21]. In the context of legal requirements, implementation order should be influenced by legal domain knowledge [19]. Legal requirements prioritization allows software engineers to focus on requirements that have legal implications early in the engineering process, avoid expensive refactoring, and demonstrate legal due diligence by incorporating laws and regulations efficiently throughout the engineering effort.

In this paper, we propose a technique for legal requirements prioritization. Our technique consists of two steps: (1) uncovering legal implications and (2) calculating a prioritization score. This technique uses numeral assignment rather than pair-wise comparisons to reduce engineer training and improve automation. We apply our technique to the iTrust Medical Records System, an open-source EHR system designed by students at North Carolina State University over nine semester-long courses for use in the United States healthcare industry. The iTrust developers expressed their requirements as Unified Modeling Language use cases in consultation with both a practicing physician and a professional from the North Carolina Healthcare Information and Communications Alliance.

2. Related Work

In this section we discuss important related work in requirements prioritization and legal requirements.

[1] Pub. L. No. 111-5. (2009)
[2] Pub. L. No. 104-191, 110 Stat. 1936 (1996)

2009 Second International Workshop on Requirements Engineering and Law (relaw'09)
978-0-7695-4102-0/10 $26.00 © 2010

2.1. Requirements Prioritization

Herrmann et al. systematically reviewed requirements prioritization techniques based on benefit and cost predictions, noting that prioritization during requirements refinement was under-researched [9]. In addition, dependencies between requirements are often neglected when producing a prioritization [9]. Herein we address the need to prioritize during refinement and manage dependencies between legal requirements.

Karlsson classifies two primary approaches to requirements prioritization: the pair-wise comparison technique and the numeral assignment technique [10]. The Analytic Hierarchy Process (AHP) serves as a well-known example of a pair-wise technique [21]. Karlsson et al. show that pair-wise comparison techniques require substantial effort upfront because every pair of requirements must be compared with one another [12]. Much of this effort must be repeated to re-prioritize requirements as they change or evolve. In contrast to pair-wise approaches, existing numeral assignment prioritization techniques take more time and produce less reliable prioritizations than pair-wise comparison techniques [10]. Herein, we develop a numeral assignment based on legal text structure, which can be produced quickly while providing a useful legal requirements prioritization.

Lehtola et al. outline five practical challenges in requirements prioritization: (1) requirements prioritization is inherently ambiguous; (2) requirements prioritization changes based on the individuals involved; (3) requirements must be prioritized in phases (4) developers need extensive domain knowledge about customer preferences to properly prioritize requirements; and (5) prioritizations are based on a multitude of factors [13]. Section 6 explains how our techniques mitigate these challenges.

2.2. Legal Requirements

The needs to resolve ambiguities, to follow cross-references, and to master extensive domain knowledge make legal compliance particularly challenging for requirements engineers [19]. Researchers have focused on building logical models of regulation directly from actual legal texts. Barth et al. employ a first-order temporal logic approach to model laws and regulations as well as measure compliance [2]. May et al. model legal texts using an access control approach [15]. Massacci et al. take a goal-oriented modeling approach to Italian data protection laws [16]. Maxwell et al. use production rules, an artificial intelligence technique, to model regulations for requirements [18]. None of this legal requirements research addresses prioritization.

Breaux et al. developed the Frame-based Requirements Analysis Method (FBRAM) [4, 5, 6]. Although primarily intended as a methodology to build requirements that achieve complete legal compliance, FBRAM also produces a comprehensive prioritization hierarchy for an entire legal text [4, 5]. The prioritization hierarchy is used to identify which requirement takes precedence in cases of legal exceptions [4, 5]. As a result, it is more useful for proving that a set of requirements has the same priority of outcomes as a set of legal regulations than for prioritizing requirements for the purposes of software construction or maintenance.

Another important consideration for legal requirements is that legal text interpretations can change over time. Laws and regulations may be amended by administrative agencies or interpreted in judicial proceedings for years after their initial passage. Legal norms in the U.S. suggest that HIPAA regulations could be revised as often as every year [3]. Engineers using pair-wise requirements prioritization techniques must reapply those techniques whenever a legal or regulatory change occurs.

3. Prioritization Techniques for Legal Requirements

In this paper, we propose a legal requirements prioritization technique consisting of two steps: (1) uncovering legal implications and (2) calculating a prioritization score. The inputs to our technique are a legal text and a set of software requirements that must comply with that legal text. Given these inputs, our approach helps engineers divide requirements into three prioritization groups: (A) non-legal requirements, (B) legal requirements needing further refinement, and (C) implementation-ready legal requirements. The first step is designed to separate requirements with legal implications with respect to the input legal text from those without. The second step further divides legal requirements into those that need refinement and those that are ready to implement. To create this division, engineers calculate prioritization scores for each requirement based on the following: (1) number of subsections mapped – S_M; (2) number of subsections contained – S_C; (3) number of cross-references – C; and (4) number of exceptions – E. Summing the resulting numbers produces an overall prioritization score. A lower score indicates higher readiness for implementation, whereas a higher score indicates more need further refinement.

We applied our technique to the iTrust requirements. Massey et al. previously evaluated the iTrust requirements for legal compliance, and provided an initial mapping of iTrust requirements to the HIPAA regulations [17]. This evaluation produced a total of 73 requirements, of which 63 were functional and 10 were non-functional [17].

2009 Second International Workshop on Requirements Engineering and Law (relaw'09)
978-0-7695-4102-0/10 $26.00 © 2010

3.1. Uncovering Legal Implications

The first step of our technique uses a requirements set and a relevant legal text as inputs. The objective is to separate the input requirements set into two output requirements sets: those requirements with legal implications and those without. Because legal texts are hierarchically structured documents requirements may be mapped at one or more levels within the hierarchy. Requirements engineers, in consultation with legal domain experts, can map a requirement to a particular subsection of a legal text. Requirements mappings are determined by finding the lowest level subsection(s) of a legal text that describes all aspects of the requirement. For example, consider the following iTrust requirement descriptions:

R-12: iTrust shall support user authentication by allowing a user to input their user ID and password.

R-23: iTrust shall record the user ID, operation, and date/time of all completed operations performed by a user, using their authorized account, during a user session.

Figure 1: iTrust Requirements 12 and 23

These requirements map to different levels of detail in the HIPAA regulations. R-12 maps to §§ 164.308(a)(3)(ii)(A) (shown below in Figure 2), 164.308(a)(4)(ii)(B), 164.312(a)(1), and 164.312(e)(1). Each of these HIPAA subsections discusses concepts relating to user authentication and encapsulated at the same level of detail in R-12.

§ 164.308(a)(3)(ii)(A) Authorization and/or supervision (Addressable). Implement procedures for the authorization and/or supervision of workforce members who work with electronic protected health information or in locations where it might be accessed.

Figure 2: Sample HIPAA Section

R-23 maps to a single high-level section: § 164.528. Section 164.528 has many subsections that describe the conditions under which individuals should be provided an accounting of disclosures, but these subsections are more specific than the details described by R-23. As a result, disclosure accounting as a whole is considered an operation that could be logged according to R-23.

Since our technique only prioritizes for legal implementation readiness, requirements engineers may wish to prioritize requirements based on additional engineering criteria. For the purposes of this study, we chose to adopt the discrete scale ranking (Low, Medium, High, and Critical) previously employed by Massey et al. [17]. Because this discrete scale does not require pair-wise comparisons, the number of prioritization decisions the requirements engineer must make is exactly equal to the number of requirements in the system. However, engineers may choose to use a more detailed prioritization technique, such as AHP, to prioritize requirements based on non-legal criteria, which is outside the scope of our work.

3.2. Prioritizing Requirements with Legal Implications

The second step of our technique, which depends on the first, calculates a prioritization score in four categories: (1) Number of Subsections Mapped – S_M, (2) Number of Subsections Contained – S_C, (3) Number of Cross-References – C, and (4) Number of Exceptions – E. Each of these categories represents an approximation of the legal complexity required to implement the requirement. Summing them forms the prioritization score (P) as shown below:

$$P = \sum_{R=1}^{n} (S_M(R) + C(R) + E(R) + S_C(R))$$

We now discuss each element of this calculation. The requirements engineer first computes the number of subsections mapped (S_M) from each requirement to the relevant elements of legal text. Consider the examples from Figure 1. R-12 has an S_M calculation of four, and R-23 has an S_M calculation of one.

Second, the requirements engineer sums the number of subsections contained (S_C) by the subsection to which the requirement maps. Section 164.528, to which R-23 is mapped, has four direct subsections: (a), (b), (c), and (d). Subsections (a) through (d) in turn have a total of 12 subsections. To calculate the total, recursively count each subsection until all subsections are counted once. Sum the result to calculate the total number of subsections contained. For § 164.528, this turns out to be 47 total subsections contained.

Third, requirements engineer determines the number of cross-references mapped (C). A cross-reference, for the purposes of this calculation, is an instance in a legal text that refers either to another section of the same legal text (internal cross-reference) or to a section of another legal text (external cross-reference). Each counts as a single cross-reference for the purposes of this calculation.

Finally, the engineer determines the number of exceptions (E). An exception is a condition specified in a legal text under which a previous statement does not hold. For example, the structure of HIPAA § 164.528(a)(1) is similar to a case statement in a procedural programming language. Each subsection is a separate exception and should be counted as such for the purposes of this calculation.

To generate the final prioritization score for a requirement, sum the results from each of the four calculations. For example, iTrust requirement R-12, found in Figure 1, maps to four different HIPAA subsections (S_M), which contain four subsections (S_C), has one cross reference (C), and has zero exceptions

978-0-7695-4102-0/10 $26.00 © 2010

(E). Thus, the prioritization score for iTrust R-12 is 4 + 4 + 1 + 0 = 9. In contrast, iTrust requirement R-23, which is also found in Figure 1, maps to one HIPAA section (S_M), which contains 47 different subsections (S_C), has 28 cross references (C), and has 11 exceptions (E). As a result, R-23 has a prioritization score of 87. Note that there are no dependencies between requirements when calculating prioritization scores as there are with a pair-wise technique.

Once each requirement is scored, the requirements engineer can separate the legal requirements into two sets. (Note that any non-legal requirements form a third set.) The first set consists of those legal requirements that are legally ready to implement and are identified by low prioritization scores. For many existing information systems that must be brought into legal compliance, legal implementation readiness is the only relevant factor that must be taken into account before implementation or refactoring takes place. However, for new or developing systems there may be additional, non-legal engineering concerns that should be taken into account before implementation for requirements in this group begins. Thus, in these situations being legally ready to implement is a necessary but not sufficient characteristic that must apply to each legal requirement prior to implementation.

The second set consists of those legal requirements that need further refinement through elicitation sessions with legal domain experts and extraction sessions with legal texts. A high prioritization score distinguishes requirements in this set. These requirements also typically map to a single high-level section number and have a comparatively large number of subsections (e.g. iTrust R-23 mapping to § 164.528 and containing 47 subsections).

In our case study, we found statistically significant differences between the low prioritization scores identifying requirements that are implementation-ready and the high prioritization scores identifying requirements that need refinement. The scope of our case study does not allow us to conclude that statistically significant differences will exist for every set of requirements analyzed on any piece of legislation. However, the lowest possible prioritization score is one, and we believe it is reasonable to implement requirements with single digit prioritization scores. We discuss our mappings, prioritization scores, and case study results in more detail in the next section.

4. Case Study Results

Initially, two requirements engineers worked together to map the requirements to their corresponding legal subsection. Although neither individual is a lawyer, both have extensive experience working with the HIPAA regulations and have previously consulted with legal domain experts to better understand HIPAA. We identified legal mappings for 46 of the 63 functional requirements in iTrust to subsections in HIPAA.

Once the legal mappings were identified, two requirements engineers, one of whom had two years of legal training, manually calculated the prioritization scores. To protect against human error, we independently counted the statistics and then averaged the final calculations to determine a final prioritization score. Figure 3, below, displays the results of this effort. Recall that lower prioritization scores indicate better readiness for implementation, whereas higher prioritization scores indicate requirements needing further refinement.

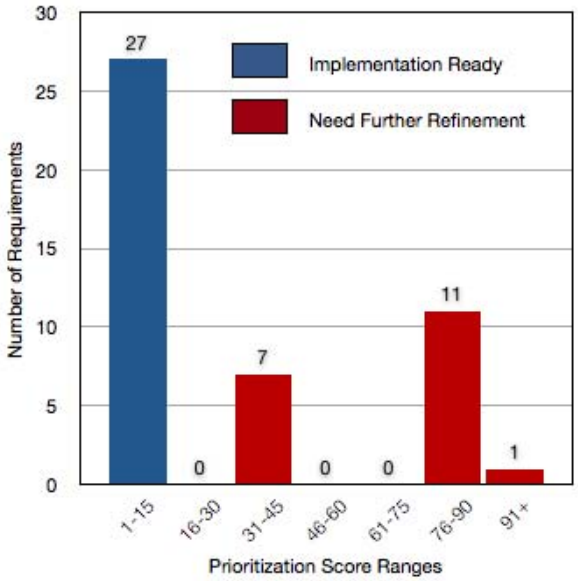

Figure 3: Average Requirements Prioritization

Our manual calculations were identical for 19 of the 46 requirements. The standard deviation for all calculated prioritization scores was 2.59 points. To visualize the prioritization scores, we grouped the results in seven 15-point score ranges. The standard deviation for the 27 requirements scoring between 1 and 15 was 2.53. The standard deviation for the 7 requirements scoring between 31 and 45 was 2.04. The standard deviation for the 11 requirements scoring between 76 and 90 was 1.66. In addition, the Fleiss' kappa inter-rater reliability, which is a statistical measure for agreement between two independent raters, for grouping these 46 requirements according to the seven groups was 0.963, which indicates an extremely high level of agreement.

The lowest scoring range displayed in Figure 3 consisted of a group of 27 requirements that we consider legally implementation-ready. Because iTrust

978-0-7695-4102-0/10 $26.00 © 2010

is an existing system being refactored to comply with HIPAA, our legal requirements prioritization is sufficient and these 27 requirements can be implemented immediately. If iTrust were a new system under initial development and there was no prioritization of requirements based on engineering criteria, then further prioritization may be needed prior to implementation because our legal prioritization technique only demonstrates that the requirements are legally ready to implement.

The 19 remaining requirements need further refinement to become legally ready to implement. The seven requirements with average prioritization scores between 31 and 45 are not easily implementable without some refinement. Consider requirement R-6 from Figure 4 below:

> **R-6:** iTrust shall maintain a patient's records for a period of seven years after their record has been disabled.

Figure 4: iTrust Requirement R-6

The HIPAA regulations provide patients the ability to request information about their record for up to six years from the date of the request. In addition, the regulations describe the conditions for recording and responding to such requests in several subsections, and those conditions are not clearly articulated in this requirement. These minor differences could cause legal complications after implementation. For example, the overhead of maintaining an additional year of patient records may hinder the ability of the healthcare facility to respond to requests for information in a timely fashion. In fact, a developer implementing this requirement has no information about how the system should respond to patient requests for information because R-6 makes no mention of this legal process described by the subsections it contains.

Twelve requirements had average prioritization scores above 76. These requirements need significant refinement before implementation. For example, consider again R-23 from Figure 1, which has an average prioritization score of 89.5. From an engineering perspective this requirement appears implementable. It provides specifics about recording data, such as the user ID, operation, date, and time, to be recorded during a user session. However, HIPAA § 164.528 describes in detail the information that must be recorded for the purposes of providing an accounting of disclosures. R-23 misses information that must be recorded, such as the purpose for an operation [§ 164.528(b)(2)(iv)] and situations under which access to this information should not be provided [§§ 164.528(a)(1)(i)-(ix)].

It is worth noting that the prioritization score ranges discovered in this study may not hold for legal domains other than HIPAA. We plan to conduct additional case studies to determine more precise scoring ranges.

5. Threats to Validity

Our case study is an exploratory case study. We developed our prioritization technique by conducting a case study in which we analyzed the iTrust requirements. Internal validity is not a concern for exploratory case studies [23]. Thus, we only discuss threats from construct validity, external validity, and reliability.

Construct validity refers to the appropriateness and accuracy of the measures and metrics used for the concepts studied [23]. This case study relies on a single source of requirements: the iTrust Medical Records System. In this study we addressed construct validity by strictly adhering to the technique as described in Section 3 and by submitting draft reports of this work for review to our colleagues at The Privacy Place and North Carolina State University.

External validity refers to the ability to generalize the findings of a case study to other domains [23]. Although no healthcare provider is currently using iTrust in industry, healthcare professionals consulted with the developers of iTrust from the beginning [17]. Similarly, although HIPAA represents a single legal domain, we have consulted with legal experts and determined that HIPAA is significantly similar in form to other regulations.

Reliability refers to the ability of other researchers to repeat a case study [23]. After developing the prioritization procedures for this study, two of the authors independently applied the techniques and afterwards compared their results to address reliability. Independently applying the techniques forced us to follow the procedures as strictly as possible. A comparison of the similarity of our results revealed a statistically significant level of agreement.

6. Summary and Future Work

This paper proposes a numeral assignment technique for prioritizing legal requirements. This technique can be used by new or existing systems that must comply with laws or regulations to prioritize the requirements relating to those laws and regulations. We focused on EHR systems, which must comply with HIPAA. In particular, we applied our technique to the 63 functional requirements defined for iTrust, an open-source EHR system. Our findings show that the iTrust requirements had 17 requirements with no mapping to a relevant element of the legal text, 19 requirements needing further refinement, and 27 requirements that are legally implementation-ready.

This work addresses several research needs identified in section 2. First, our prioritization

978-0-7695-4102-0/10 $26.00 © 2010

technique addresses requirements refinement and dependency management, previously identified as an under-researched area by Herrmann et al. [9], by focusing efforts on legally implementation-ready requirements. In addition, this work demonstrates that numeral assignment prioritization techniques may be useful in the legal domain. This numeral assignment technique provides time- and effort-savings when laws, regulations, or the requirements change since only the affected requirements need to have their prioritization scores recalculated. Finally, given a requirements mapping and a structured legal text, our technique can mitigate the five practical challenges identified by Lehtola et al. [13] by: (1) limiting ambiguity to the requirements mapping, (2) calculating prioritizations based on the structure of a legal text rather than individual opinion, (3) reapplying the technique to requirements previously found to need refinement, (4) limiting domain knowledge to the requirements mapping, and (5) explicitly definite the factors involved in requirements prioritization.

We plan to conduct additional case studies to further validate our techniques by involving more software engineers, comparing our approach with other prioritization techniques, and applying our technique to other legal domains. Also, we are in the process of implementing tool support to improve the speed and accuracy of calculating prioritization scores, to investigate additional weightings for elements in the prioritization score calculation, and to further reduce dependency on individual interpretations.

Acknowledgements

This work was partially supported by NSF ITR Grant #522931 and NSF Cyber Trust Grant #0430166. The authors would like to thank Jeremy Maxwell for his help mapping iTrust requirements to HIPAA.

References

[1] P. Avesani, C. Bazzanella, A. Perini, and A. Susi. Facing scalability issues in requirements prioritization with machine learning techniques. 13th IEEE Intl. Conf. on Req. Eng., pp 297–305, Aug.-2 Sept. 2005.

[2] A. Barth, A. Datta, J. C. Mitchell, and H. Nissenbaum, "Privacy and Contextual Integrity: Framework and Applications," Proc. of the 2006 IEEE Symposium on Security and Privacy, pp.184-198, 2006.

[3] K. Beaver and R. Herold, The Practical Guide to HIPAA Privacy and Security Compliance. Auerbach Publications, 2004.

[4] T. D. Breaux. Legal Requirements Acquisition for the Specification of Legally Compliant Information Systems. PhD Dissertation, North Carolina State University, 2009.

[5] T. D. Breaux and A. I. Antón, "Analyzing Regulatory Rules for Privacy and Security Requirements," IEEE Trans. on Soft. Eng., 34(1), pp. 5–20, Jan. 2008.

[6] T. D. Breaux, M. W. Vail, and A. I. Antón, "Towards Regulatory Compliance: Extracting Rights and Obligations to Align Requirements with Regulations," 14th IEEE Intl. Req. Eng. Conf. pp. 49–58, Sept. 2006.

[7] President George W. Bush, "Transforming Health Care: The President's Health Information Technology Plan," archives.gov, accessed 7 Feb. 2009.

[8] J. Cleland-Huang and M. Denne. Financially informed requirements prioritization. 27th Intl. Conf. on Soft. Eng., pp. 710–711, May 2005.

[9] A. Herrmann and M. Daneva. Requirements prioritization based on benefit and cost prediction: An agenda for future research. 16th IEEE Intl. Req. Eng., Sept. 2008.

[10] J. Karlsson. Towards a Strategy for Software Requirements Selection. Licentiate Thesis 513, Department of Computer and Information Science, Linköping University, Sweden, 1995.

[10] J. Karlsson. Software requirements prioritizing. 2nd Intl. Conf. on Req. Eng., pp. 110–116, Apr. 1996.

[11] J. Karlsson and K. Ryan. A cost-value approach for prioritizing requirements. IEEE Software, 14(5):67–74, Sep/Oct 1997.

[12] J. Karlsson, C. Wohlin, and B. Regnell. An evaluation of methods for prioritizing software requirements. Information and Software Technology, 39(14-15), 1998.

[13] L. Lehtola, M. Kauppinen, and S. Kujala. Requirements prioritization challenges in practice. Product Focused Software Process Improvement, pp. 497–508, 2004.

[14] D. Manos, Obama: EHRs for Americans by 2014. Healthcare IT News, January 8, 2009.

[15] M. J. May, C. A. Gunter, and I. Lee, "Privacy APIs: Access Control Techniques to Analyze and Verify Legal Privacy Policies," 19th IEEE Computer Security Foundations Workshop, pp. 85 – 97, 2006.

[16] F. Massacci, M. Prest, and N. Zannone, "Using a Security Requirements Engineering Methodology in Practice: The Compliance with the Italian Data Protection Legislation," Computer Standards & Interfaces, 27(5), pp. 445–455, 2005.

[17] A. K. Massey, P. N. Otto, L. J. Hayward, and A. I. Antón, "Evaluating Existing Security and Privacy Requirements for Legal Compliance." (In Press for) Req. Eng. Journal, Springer Verlag, 2009.

[18] J.C. Maxwell, A.I. Antón, "Developing Production Rule Models to Aid in Acquiring Requirements from Legal Texts", 17th IEEE Intl. Req. Eng. Conf., 2009.

[19] Paul. N. Otto and Annie. I. Antón, "Addressing Legal Requirements in Requirements Engineering," 15th IEEE Intl. Req. Eng. Conf., pp. 5–14, 15-19 Oct. 2007.

[20] A. Perini, A. Susi, F. Ricca, and C. Bazzanella. An empirical study to compare the accuracy of AHP and cbranking techniques for requirements prioritization. 5th Intl. Workshop on Comparative Evaluation in Requirements Engineering. pp. 23–35, Oct. 2007.

[21] T. L. Saaty, The Analytic Hierarchy Process. McGraw-Hill, 1980.

[22] E. Singer. A big stimulus boost for electronic health records. MIT Technology Review, February 20, 2009.

[23] R. K. Yin, Case Study Research: Design and Methods, vol. 5 of Applied Social Research Methods Series. Sage Publications, 3rd ed., 2003.