Towards Compliant Reference Architectures by Finding Analogies and Overlaps in Compliance Regulations

Eduardo B. Fernandez and Dereje Yimam
Dept. of Computer Science and Engineering, Florida Atlantic University, 777 Glades Rd, Boca Raton, Florida, U.S.A.

Keywords: Compliance, Regulations, Business Software, Security Patterns, Reference Architectures.

Abstract: Business software is subject to a variety of regulations depending on the type of application. For example, software handling of medical records must follow HIPAA; software for financial applications must comply with Sarbanes Oxley, and so on. A close examination of the policies included in those regulations shows that they have analog and common aspects. Analog parts of regulations can be expressed as Semantic Analysis Patterns (SAPs), which can lead to building similar parts in other regulations. Overlapping parts usually correspond to security patterns and can be used to add security to other regulations. If we collect SAPs and security patterns in a catalog we can build reference architectures (RAs) for existing and new regulations. The resultant Compliant RAs (CRAs) can be used as guidelines for building compliant applications.

1 INTRODUCTION

In many countries, business activities have government, state, or industry-based regulations. Regulations are sets of policies about how the information used in some areas of business must be handled, i.e., software used in those businesses must comply with these regulations. Some laws state that organizations are responsible for all compliance-related issues. The cost of not being compliant may result in penalty fees, possible lawsuits, and bad business reputation. Compliance implies keeping a set of rules that implement the policies defined in the regulations, which are then enforced by the software when handling the corresponding type of data. In the opinion of (Massey et al., 2011), legal compliance may become the most important Non-Functional Requirement (NFR) for a large number of software systems. Government and state regulations are mandatory while industry regulations are suggestions. However, not following industry regulations may hurt the marketing possibilities of a software system.

Regulations are written by lawyers and usually are lengthy, hard to read, at times redundant, perhaps ambiguous, and maybe even inconsistent at some places. Incorrect or imprecise implementations of regulations may lead to lawsuits and may harm people. (Massey et al., 2011) reports that most computer science graduate students have trouble understanding regulations. To make regulations clearer and more precise there has been attempts to analyze them to understand the rights and obligations of the participants (Breaux and Anton, 2008; Lam and Mitchell, 2008). However, there has been few attempts to make clear the software architecture required for the implementation of the policies in the regulations. We can express regulations in the form of patterns; as shown for HIPAA in (Fernandez and Mujica, 2014a). A pattern is a solution to a recurrent problem in a given context (Gamma et al., 1994). Several methodologies exist for building secure systems using patterns (Uzunov et al., 2015); these methodologies could also handle compliance.

The specific regulations to be followed depend on the type of application. For example, software handling medical records must follow HIPAA (HIPAA, 2013); software for financial applications must comply with Sarbanes Oxley (SOX, 2015), and so on. Some applications may need to follow more than one regulation. A close examination of the policies included in those regulations shows that they have analogies. By that, we mean that portions of the regulations handle information in a similar way. Different regulations also have straight commonalities, e.g., they specify the same enforcement mechanism. We show here that by identifying analogies and commonalities we can make regulations much clearer and easier to

B. Fernandez E. and Yimam D..
Towards Compliant Reference Architectures by Finding Analogies and Overlaps in Compliance Regulations.
DOI: 10.5220/0005575604350440
In *Proceedings of the 12th International Conference on Security and Cryptography* (SECRYPT-2015), pages 435-440
ISBN: 978-989-758-117-5

implement. If we collect these aspects as patterns in a catalog, we can build reference architectures (RAs) for existing as well as new regulations. The resultant RAs can be used as guidelines for building compliant applications. We can also use this catalog to complement secure or compliant software development methodologies. We can make the software developed using one of the methodologies mentioned above not just secure but also compliant if we add to them a catalog of patterns that describe the regulation(s). In addition, incorporating regulations described as patterns into reference architectures we can generate applications that comply with these regulations (Fernandez and Mujica, 2014b). We describe our models using the Unified Modelling Language (UML) (Rumbaugh et al., 1999), at times enhanced using the Object Constraint Language (OCL) (Warmer and Kleppe, 2003).

Specifically, our contributions include a demonstration that regulations have analog and common aspects that can be leveraged to build or enhance RAs compliant with different regulations and we show an example of how to do this.

Section 2 describes some background about regulations and patterns. We show in Section 3 that some regulations perform similar actions with their data and we can deduce patterns by analogy. Section 4 indicates commonalities between regulations. Section 5 considers related work. We end with some conclusions in Section 6.

2 BACKGROUND

2.1 Regulations and Standards

We summarize below four of the most common regulations in the US, which we use as examples in Section 4. These regulations apply to a large variety of common and /or important business applications.

HIPAA (Federal regulation): Healthcare organizations are required to comply with the Healthcare Insurance Portability and Accountability Act (HIPAA), intended to protect individual health information. It requires covered entities (i.e. health care providers) to protect the privacy of the patient's Protected Health Information (PHI) (HIPAA, 2015), for which providers must use data encryption, authentication, authorization, backup, monitoring, notification, and disaster recovery. Also, providers and transactions must have unique identifiers.

PCI-DSS (Credit Card industry regulation): Companies that handle cardholder information are required to comply with the Payment Card Industry Data Security Standard (PCI DSS). Cardholder information includes debit, credit, prepaid, e-purse, ATM, and Point of Sale (POS) cards (PCI, 2015). PCI-DSS require using data encryption, access control, auditing, disaster recovery, monitoring, and notification.

Sarbanes-Oxley Act (SOX) (Federal regulation): SOX establishes standards for all US publicly-traded companies in order to protect shareholders and the general public from accounting errors and fraudulent practices in the enterprise (SOX, 2015). SOX enforces control on user management, system development, program and infrastructure management, monitoring, backup, auditing, and disaster recovery.

Gramm-Leach-Bliley Act (GLBA) (Federal regulation). It requires financial institutions that offer financial products or services to consumers to develop, implement, and maintain a comprehensive information security program that protects the confidentiality and integrity of customer records (GLBA, 2015).

2.2 Patterns and Related Concepts

As indicated, a (software) pattern is a solution to a recurrent problem in a given context. A pattern embodies the knowledge and experience of software developers and can be reused in new applications. Analysis patterns can be used to build conceptual models (Fernandez and Yuan, 2000; Fowler, 1997), design and architectural patterns are used to build flexible software (Buschmann and Meunier, 1996) (Gamma et al., 1994), and security patterns can be used to build secure systems (Fernandez et al., 2006; Fernandez, 2013). The concept of Semantic Analysis Pattern (SAP) is an analysis pattern realizing a small set of related use cases; it will be used here as a conceptual unit. Well-thought patterns implicitly apply good design principles. Pattern solutions are usually expressed using UML diagrams, semi-formal solutions that can be more formalized using OCL (Warmer and Kleppe, 2003). The typical solution provided by a pattern comes in the form of a class diagram complemented with some sequence diagrams and possibly activity or state diagrams. This level of detail and precision allows designers to use them as guidelines and users to understand the effect of the mechanisms they represent. Patterns are also good for communication between designers and to evaluate and reengineer existing systems. Design patterns are already widely accepted in industry; Microsoft, Siemens, Sun, and IBM, among others, have web pages and even books about them. Security

patterns are starting to be accepted, with some companies producing catalogs in books and in the web, including IBM, Microsoft, Amazon, and Sun (Oracle). We have written a large amount of security patterns, some of which (about 70), are described in (Fernandez, 2013).

A *Domain Model* (DM) is a model of an area of knowledge, e.g., electrical engineering. We can think of a DM as a compound pattern, including several simpler patterns that represent specific aspects of the domain. A *Reference Architecture* (RA) is a generic architecture, valid for a particular domain (or set of domains), with no implementation aspects (Avgeriou, 2003; Taylor et al., 2010). It is reusable, extendable, and configurable, that is, it is a kind of pattern for whole architectures and it can be instantiated into a specific software architecture by adding implementation-oriented aspects.

3 ANALOGY

Typically, regulations refer to four aspects: data with indications or classifications of their sensitivity; the entities (stakeholders) involved in handling this data, usually defined by their roles; the rights of these roles with respect to the data; and the obligations of the roles when they access data. Keeping in mind these four aspects we can see that some regulations have parallel concepts. Analogy to discover new SAPs is applied in (Fernandez and Yuan, 2000). Analogy implies the realization that the information in a specific model is handled in a similar way in another model; i.e., the other model has the same concepts (possibly with different names) related in a similar way and followed by specialization.

Figure 1 shows a UML class model showing parts of the HIPAA rules. The role Patient has a Medical Record for which the role has the right to read it and authorize its use. Medical Records include Treatment Histories. The role Doctor has the right to read and modify the records of his own patients. Medical records are related to each other based on some Medical Relationship that relates records of contact for infectious diseases or genetic relationships. Reading of medical records by external entities requires patient notification. The left side of Figure 1 can be considered a SAP describing the rights of patients and doctors as well as a system obligation. The classes Health Care Provider and Patient on the right side of Figure 1 are in fact part of another pattern describing two of the stakeholders of the regulation (Sorgente and Fernandez, 2004).

Figure 2 shows a SOX model obtained from the HIPAA model by making the following analogies: Patient=>Investor, Medical Record=>Financial Record, Doctor=>Broker, Treatment History=>Financial Account. The left side of the figure is the analog of the medical record SAP and can be derived directly from it. As in all patterns, it is not a plug-in but it needs to be tailored. A type of tailoring is shown in Figure 2 where the OCL constraints of Figure 1 have been expressed in words and the class names reflect the different context. We can carry this analogy to any regulation that requires handling of some type of records that belong to individuals.

The model of Figure 1 requires a platform that can apply content-dependent restrictions and by using again analogy we know that this type of restrictions will also be needed in the model of Figure 2. We can generalize the patterns of Figures 1 and 2 and define an abstract Record Protection pattern from which we can derive patterns specific to new regulations.

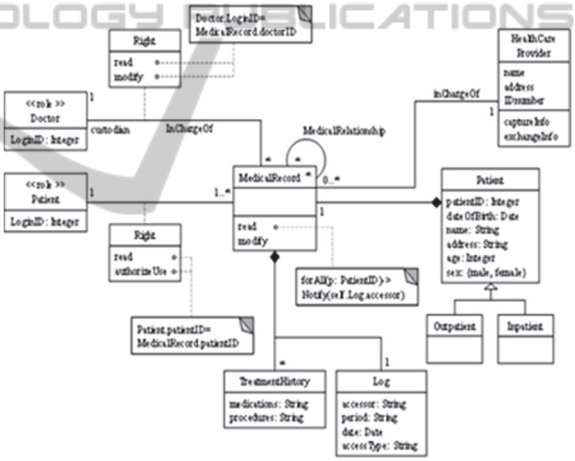

Figure 1: A partial model of HIPAA.

4 OVERLAP

Reading the descriptions of the regulations in Section 2 we can see that specific security mechanisms appear in most of them because they require protection of information. Compliance requires attributes such as confidentiality, integrity, availability, reliability and accountability. As a result, there are a number of commonalities among regulations and standards that could be abstracted as patterns. Some of the commonalities are described below.

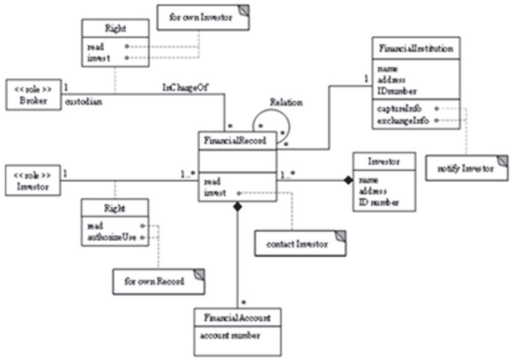

Figure 2: A model for parts of SOX.

For example, privacy requirements are present in many regulations. Privacy requires the use of security mechanisms that can be described by security patterns. For example, a privacy policy that indicates that "hospital patients can see their own medical records" requires content-dependent access control, which can be enforced by a corresponding security pattern (Fernandez et al., 2014). Obligations can be realized as part of authorization rules but also can be defined as separate rules. All regulations require participants to be uniquely identified. Most of the regulations described here show the following requirements:

1) **Security:** policies and procedures to regulate the security of data, systems, applications and configurations. Security implies authentication, authorization, and encryption which provide confidentiality and integrity. Security management focuses on policies and procedures to create, modify, delete, and review user access control, security settings and configurations.
2) **Privacy:** policies and procedures to regulate the use and the disclosure of sensitive information that uniquely identify an individual or client.
3) **Logging and Audit Control:** policies and procedures to record and examine activities of users (data accessed, time), configurations of systems and applications.
4) **Secure Data Transmission and Storage:** policies and procedures to secure data transmission and storage (cryptography and digital signatures for transmission, authentication and authorization for storage).
5) **Notification:** when information is accessed by entities not specifically entitled to do so, users must be notified.
6) **Reporting:** policies and procedures to generate an incident report to assure compliance, including security breaches and compliance on user activities, system activities, and configurations changes.
7) **Compliance Monitoring:** Monitors and enforce compliance by applying compliance rules and regulations.
8) **Compliance Analysis:** Analyze the overall activities of users, data, configurations, systems and applications by auditing logs and records.
9) **Backup:** policies and procedures to archive protected data to ensure recovery.
10) **Disaster Recovery:** policies and procedures to recover systems, applications, data and configurations.
11) **Sanitation:** policies and procedures to sanitize storage devices when they are out of use.
12) **Emergency Access:** policies and procedure to access protected data in the case of emergency.

We have patterns for most of these. For those aspects where there are no patterns, we list them as future work and include in the list: reporting, compliance monitoring, compliance analysis, disaster recovery, sanitation, and emergency access.

In spite of the fact that regulations explicitly indicate the types of security mechanisms they require for their protected information, the truth is that the system must be secure as a whole. It doesn't really matter if private information is disclosed through a path not considered in the corresponding regulation, the record keeper is still legally responsible. In other words, the list of the security mechanisms indicated by the regulation should be interpreted mostly as a core and not as a complete set of requirements, what matters is avoiding misuse of the information.

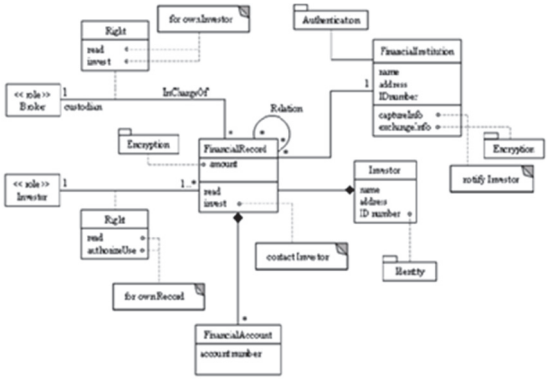

Figure 3: A model for secured SOX.

If we catalog the patterns, we can build models for other regulations starting from the models we have. For example, we can build an RA for SOX starting from the HIPAA RA. Using analogy we could have

deduced the model of Figure 2 and using the table of commonalities we can add corresponding mechanisms in the form of patterns. Figure 3 shows the model of Figure 1 with the addition of mechanisms for Encryption, Authentication, and Reporting. Note that Figure 2 already had Authorization in the form of RBAC.

5 RELATED WORK

(Breaux and Anton, 2008) presented a methodology for extracting formal descriptions of rules and regulations. In particular, they analyzed a method for acquiring and presenting data access requirements and a method for defining priorities on these requirements. We work at a higher architectural level, considering computational units, expressed as patterns.

Another attempt to formalize HIPAA is given in (Lam et al., 2009). The authors used a restricted version of Prolog and analyzed parts of HIPAA. They found some conflicts in its rules and also implemented a compliance checker.

(Massacci et al., 2005) used the Secure Tropos requirements engineering modeler to verify that the University of Trento complied with the Italian laws on privacy and data protection.

(Hamdaqa and Hamou-Lhadj, 2009) built a citation graph that can be used by analysts to navigate through the provisions of various interrelated laws, to uncover overlaps and possible conflicts or to simply understand specific compliance documents. The author also used a Compliance Decision Support System (CompDSS) to identify compliance similarities and differences. In particular, the author used citation graphs to uncover overlapping and possible conflicts, detecting important provisions by ranking, assessing the impact of change in a particular act, and checking its consistency. The author focused only in HIPAA, SOX and GLBA standards. This is the only paper we know which took advantage of overlaps in regulations.

All this work focuses on the correct and complete interpretation of the regulation rules. As far as we know, there has been almost no work on the system architecture aspects of the implementation of the regulations; in particular, nobody else has applied patterns to describe regulations. In (Fernandez et al., 2006) and (Fernandez, 2013) a methodology to build secure systems is proposed. This was extended in (Uzunov et al., 2015). It could be possible to tailor some of its steps to consider the use of regulation patterns and we leave this as future work.

Note that patterns are not components or software modules, they are suggestions expressed as models or even words that represent specific software artifacts; they can be instantiated once or many times in the same application. As validation for our ideas we cannot implement patterns or build a complete catalog that by necessity must be open ended. What we can do though, is to show a complete example of the use of our ideas in deriving part of an RA for a new regulation. We cannot do this here for lack of space, however.

6 CONCLUSIONS

Companies that develop software to be used by institutions that must follow a variety of regulations can benefit by first building a catalog of patterns that can be used as building blocks to build complete regulation models. These patterns can describe specific policies or security mechanisms that appear in several of the regulations. We can also build a catalog of patterns based on making analogies across regulations as the ones shown in Section 4, which can be used for building support for new regulations. The use of these catalogs leads to a factory for RAs from where we can derive applications that comply with one or several regulations. The fact that patterns are not components but abstract templates makes their use easily shareable by software companies, one catalog is enough from which each company can produce different implementations as components, web services, or ad hoc modules.

We are working on a structure to automatically derive compliant applications starting from compliant RAs. For example, the patterns of Figure 1 could be the start of a RA for HIPAA, where we can add patterns for billing, lab tests, diagnostics, and other medical procedures all of which would be compliant. From such an RA we can derive concrete architectures such as a cloud-based RA for HIPAA. This is future work, to which we can add writing the missing patterns identified in Section 4. Our results are also useful to decide if a legacy system is or can be made compliant with specific regulations or to evaluate if a new platform can support the regulations supported by the old platform.

REFERENCES

Avgeriou, P. 2003, 'Describing, instantiating and evaluating a reference architecture: A case study', *Enterprise Architecture Journal.*

Breaux, T. D. and Anton, A. I. 2008, 'Analyzing regulatory rules for privacy and security requirements', *IEEE Trans.* on Soft. Eng., vol. 34, No 1, Jan. /Feb., 5-20.

Breaux, T.D. and Gordon, D.G. 2011, 'Regulatory requirements as open systems: Structures, patterns and metrics for the design of formal requirements specifications', Rept. CMU-ISR-11-100, Carnegie Mellon University.

Buschmann, F., Meunier, R., Rohnert, H., Sommerlad, P., Stal, M. 1996, *Pattern-Oriented Software Architecture: A System of Patterns*, Volume 1, Wiley.

Fernandez, E. B., Larrondo-Petrie, M.M., Sorgente, T., and VanHilst, M. 2006, 'A methodology to develop secure systems using patterns", Chapter 5 in *"Integrating security and software engineering: Advances and future vision'*, H. Mouratidis and P. Giorgini (Eds.), IDEA Press, 107-126.

Fernandez, E. B. and Yuan, X. 2000, 'Semantic analysis patterns', *Proceedings of the 19th Int. Conf. on Conceptual Modeling*, ER2000, 183-195.

Fernandez, E. B. 2013, *Security patterns in practice: Building secure architectures using software patterns*, Wiley Series on Software Design Patterns.

Fernandez, E. B. and Mujica, S. 2014, 'Two patterns for HIPAA regulations', *Procs. of AsianPLoP (Pattern Languages of Programs),* Tokyo, Japan.

Fernandez, E. B. and Mujica, Sergio 2014, 'From domain models to secure and compliant applications', *Procs. 12th LACCEI.*

Fernandez, E. B., Monge, Raul, and Hashizume, Keiko 2015, 'Building a security reference architecture for cloud systems', *Requirements Engineering*. DOI: 10.1007/s00766-014-0218-7.

Fernandez, E. B., Monge, R., Carvajal, Encina, O., Hernandez, J., and Silva, P., R. 2014, 'Patterns for Content-Dependent and Context-Enhanced Authorization'. *Proceedings of 19th European Conference on Pattern Languages of Programs*, Germany.

Fowler, M. 1997, *Analysis patterns – Reusable object models*, Addison-Wesley.

Gamma, E., Helm, R., Johnson, R., Vlissides, J. 1994, *Design Patterns: Elements of Reusable Object-Oriented Software*, Addison-Wesley, Boston, Mass.

GLBA 2015, Gramm-Leach-Bliley Act. Available from: <http://www.business.ftc.gov/privacy-and-security/ gramm-leach-bliley-act. [10 January 2015].

Hamdaqa, M. and Hamou-Lhadj, A. *2009,* 'Citation Analysis: An Approach for Facilitating the Analysis of Regulatory Compliance Documents', *Procs. 2009 6th Int. Conf. on Information technology: New Generations*, IEEE, 278-283.

HIPAA 2015, Understanding Health Information Privacy. Available from: http://www.hhs.gov/ocr/privacy/ hipaa/understanding/index.html. [8 January 2015].

HIPAA 2013, HIPAA Administrative Simplification. Available from: <http://www.hhs.gov/ocr/privacy/ hipaa/administrative/combined/hipaa-simplification-201303.pdf. [10 January 2015].

Lam, Peifung E., Mitchell, John C., Sharada Sundaram 2009, 'A Formalization of HIPAA for a Medical Messaging System', in *Trust, Privacy and Security in Digital Business*, Lecture Notes in Computer Science, Volume 5695, 73-85.

Massacci, F., Presti, M., and Zannone, N. 2005, 'Using a security requirements engineering methodology in practice: the compliance with the Italian data protection legislation', *Computer Standards & Interfaces,* 27 (5), 445-455.

Massey, A.K., Smith, B., Otto, P.N., and Anton, A.I. 2011, 'Assessing the accuracy of legal implementation readiness decisions', *19th IEEE Int. Reqs. Eng. Conf*., 207-216.

PCI 2015, Official Source of PCI DSS Data Security Standards. Available from: <https://www.pcisecurity standards.org/security_standards/ index.php>. [11 January 2015]

Rumbaugh, J., Jacobson, I., and Booch, G. 1999, *The Unified Modeling Language Reference Manual*, Addison-Wesley, Boston, Mass.

Sorgente, T. and Fernandez 2004, 'Analysis patterns for patient treatment', *Procs. of PLoP.*

SOX 2015, The Sarbanes-Oxley Act. Available from :< http://www.soxlaw.com/>. [11 January 2015].

Taylor, R. N., Medvidovic, N., and Dashofy, N. 2010, *Software architecture: Foundation, theory, and practice,* Wiley.

Uzunov, A., Fernandez, E. B., Falkner, K. 2015, 'ASE: A Comprehensive Pattern-Driven Security Methodology for Distributed Systems', *Journal of Computer Standards & Interfaces , Volume 41*, *September 2015*, *Pages 112-137, http://www.sciencedirect.com/science /article/pii/S0920548915000276*

Warmer, J. and Kleppe, A. 2003, *The Object Constraint Language* (2nd Ed.), Addison-Wesley.