Abstract - This paper describes proposed privacy extensions to UML to help software engineers to quickly visualize privacy requirements, and design privacy into big data applications. To adhere to legal requirements and/or best practices, big data applications will need to apply Privacy by Design principles and use privacy services, such as, and not limited to, anonymization, pseudonymization, security, notice on usage, and consent for usage. We extend UML tools with ribbon icons representing needed big data privacy services. We further illustrate how privacy services can be usefully embedded in use case diagrams using containers. These extensions to UML help software engineers to visually and quickly model privacy requirements in the analysis phase; this phase is the longest in any software development effort. As proof of concept, a prototype based on our privacy extensions to Microsoft Visio’s UML is created and the utility of our UML privacy extensions to the Use Case Diagram artifact is illustrated employing an IBM Watson-like commercial use case on big data in a health sector application.

Keywords: Big data applications, Privacy by Design, PhD, privacy engineering, UML extensions, privacy services, requirements analysis, software engineering, anonymization, pseudonymization, use case diagrams

I. INTRODUCTION

The digitization of vast quantities of information has led to the collaboration of many stakeholders to develop shared standards-based interoperable platforms to support the efficient analysis and flow of information to accelerate new discoveries. Further and new standards-based approaches are required to help make the output of big data algorithms more secure and protected. Generically, standards promote good system design, regulatory compliance, facilitate information interchange and interoperability, and foster innovation through multi-stakeholder collaboration. Organizations gain from the benefits of standards’ adoption and market differentiation. Standards regularly protect the consumer through reduced stranding risks and lock-in effects. The standards creation process itself can spur the emergence of new technologies. New and upcoming privacy standards are expected to foster another level of protection altogether to personal data in big data settings.

In this paper, we specifically examine how we can aid software engineers to embed privacy in UML use case diagrams to visualize complex stakeholder interaction with software systems. These include big data-driven software platforms such as those offered by many popular companies such as Google, Flurry, and Facebook.

Software engineers use the popular software engineering modeling language, UML, to communicate on issues in large-scale systems. The Object Management Group (OMG)’s UML is an ISO software engineering industry modeling standard. Because of its ubiquity, UML can facilitate software engineers to communicate, understand, and collaborate on building software that embeds privacy by design. Indeed, UML diagram models are frequently used for sharing vision, giving visual representations of systems, or parts of systems to be built, influencing code generation, and documenting software requirements and design. UML helps to reduce efforts in authoring long documentation to describe a complex system. It abstracts away details that are confusing and makes it easier for the software developers to understand and examine a system’s behavior, data, and process models quickly in comparison to when textual documentation is used.

The idea and approach to extending the Unified Modeling Language, UML, with privacy services to help software engineers quickly learn about privacy requirements and easily visualize and embed them in their designs were first proposed in (Jutla 2012 a,b,c,d). This paper’s unique contribution is the extension of the UML use case diagram to support privacy requirements and services in the context of Big Data applications. Specifically, we motivate the privacy extensions to UML use case diagrams and focus on a popular subset of the privacy services important to Big Data contexts in Health, namely those supporting de-identification, and security. This work specifically illustrates the applicability of UML privacy extensions to the protection of health and genetic information privacy in Big Data applications. It embeds key aspects of Privacy by Design principles, including facilitating software engineers to be proactive, not reactive, privacy as a default setting, and respect for users’ privacy.

In any software development effort, the requirements specification phase takes the longest, whether based on agile processes with API or web service composition,
adaptation of the software development life cycle processes, or waterfall model. Hence any help to speed up or automate this phase is extremely useful to application delivery. Delivery of Big Data applications with privacy embedded by design is no exception.

II. OVERVIEW OF A BIG DATA USE CASE THROUGH A PRIVACY LENS

Watson is IBM's artificial intelligence software system that achieved popular acclaim for beating top Jeopardy game experts. In February 2013, IBM announced its first large commercial-grade Watson-based application. An excerpt of its press release is given here:

"Advancing Oncology Through Evidence Based Medicine. To date, Watson has ingested more than 600,000 pieces of medical evidence, two million pages of text from 42 medical journals and clinical trials in the area of oncology research. Watson has the power to sift through 1.5 million patient records representing decades of cancer treatment history, such as medical records and patient outcomes, and provide to physicians evidence-based treatment options all in a matter of seconds. In less than a year, Memorial Sloan-Kettering has immersed Watson in the complexities of cancer and the explosion of genetic research that has set the stage for changing care practices for many cancer patients with highly specialized treatments based on their personal genetic tumor type."

This IBM work is being done in partnership with the Memorial Sloan-Kettering Cancer Centre and WellPoint, a health insurance company. Within this scenario what would be patients privacy concerns and how can software engineers design software to ensure that patients retain privacy and benefit from advances in big data technologies? Understandably, cancer patients would give up their privacy willingly in order to maximize chances for beating the disease and to get the best possible customized care. We are advocating that users can get all the Big Data applications’ benefits, and have any level of individual privacy that they require, if software engineers thoughtfully and methodically embed privacy by design in the software systems. This is the positive sum paradigm, one of the 7 principles of Privacy by Design (Cavoukian, 2013). Commissioner Ann Cavoukian and IBM researcher Jeff Jonas (2012) identified key Privacy by Design features for Big Data, including attribution of data, data tethering, analyses on anonymized data, secure audit logs, favoring of false negatives over false positives, self-correcting false positives, and information transfer accounting. To contain the scope of this paper, we focus on visualizing de-identification requirements analysis here.

Consider genetic privacy in the context of health applications, and a future scenario where a system, such as Watson, could consume individual genome maps and advanced personalized medicine. Patients may be concerned that the insurance company could learn of a gene they carry. The implications of having the gene may cause discriminatory practices in job hiring, promotion, education, housing, or health and life insurance. While less of a concern for US citizens since the passing of the 2008 Genetic Information Non-Discrimination Act (GINA), other countries’ legal systems are not as progressive in this area. Canadians have little genetic privacy protection beyond human rights laws that impose onerous processes on citizens to obtain justice. The British Government and the Association of British Insurers agreed on a voluntarily moratorium on insurers’ exploitation of genetic information until 2017. Britain and the European Union also have monetary limits as to when an insurer may ask for the results of genetic testing. For example, to alleviate the risk of disability payouts in excess of a certain amount, insurers may ask for genetic testing for high-salaried workers. Thus, in many countries, software engineers have ethical responsibility around building-in global genetic privacy protection into Big Data health applications. The discrepancies in genetic privacy laws may also cause patients to not share valuable data that could otherwise have led to breakthroughs. Cleverly designed big data applications that protect patient privacy can play a part in breaking such barriers to the advancement of medicine, health, and wellness.

III. STAKEHOLDERS’ PRIVACY DE-IDENTIFICATION REQUIREMENTS IN WATSON-LIKE BIG DATA USE CASE

Big data privacy protection traditionally uses anonymization and pseudonymization as general privacy services to de-identify users. They modify data, about individuals/subjects, in such a way that it prevents viewers of that data to be able to identify individual subjects. Pseudonymization (Chaum, 1981) is used to replace personally identifiable information (PII) with a pseudonym/code, which cannot be used to identify an individual, prior to the release of the data. This can be done on a per PII field/fact basis or by replacing all of the individual’s PII with one pseudonym. By keeping the mapping of a subject’s pseudonym to the subject’s PII secret from the viewers of the released data, pseudonymization strives to prevent the viewers of the data to identify individuals, while the original data can be still reconstructed by using the mapping of pseudonyms to PII. The Committee on Human Research Definitions, at the University of California San Francisco, defines pseudonymized data as indirectly identifiable2. It also defines de-identified/anonymized data to contain no link

---


that can be used to link the de-identified data back to the specific subjects. The HIPAA act, which governs the use of patient’s records in the USA, defines a set of identifiers that must be removed for patient’s data to be considered as de-identified. Pseudonymous de-identification of data has been shown to be susceptible to linking attacks that use the published data set and other publicly available data to identify subjects. Thus various anonymization methods, which use generalization and removal of data, have been developed to reduce the risk of identifying subjects. As generalization and removal of data reduce the utility and quality of the data for research purposes, there is a trade-off between the research value of the published data and the reduction of the risk of identifying subjects through linkages that may exist through publicly available data sets.

Consider now, some of the key stakeholders in a Watson-like use case scenario that generates alternative treatment options for a patient and recommends the best suited one to a doctor. The implicit stakeholder in the scenario is the main data subject, the patient, and his need for privacy. Let’s further consider the data scientist and patients’ viewpoints. Both stakeholders would like as much information going into a system, such as Watson, on patient cases as possible. The data scientist cares about data quality. Obviously data anonymization techniques that replace values from random distributions can compromise quality. Thus her preference is for a pseudonymization service for de-identification. What the patient cares about is the transfer of his personal information to authorized recipient(s) of the data. The patient has different degrees of need for de-identification. If the patient’s doctor, nurse, or other players, in the patient’s direct circle of care, are the recipients of say Watson’s analysis, no de-identification is necessary for output to these recipients. But de-identification requiring various degrees of anonymization techniques may be needed for other specified and unspecified recipients. Thus, the required privacy services will be determined according to contexts involving roles, identity of third parties, types of cascading privacy controls, place, time, and so on.

We assume that the core principle of data minimization (Gurses et al, 2011) has been applied to the Big Data set to which the data scientist has access. That is, data that is not needed should not be present and/or is not collected, or included in the big data set. Additionally, data minimization techniques, such as generalization, shall be applied to the data set where possible for the health context. Informed patients recognize that in medicine, knowing gender, environmental variables, age, etc. may be important features that, if anonymized or minimized too far, could lead to less precision; hence a low-grade pseudonymity technique may be more appropriate and permissible for the internal data scientist in the Cancer Centre. Further, it is being increasingly recognized that protection at the point of usage of data is becoming more important and effective than at the point of data collection in today’s big data environment (World Economic Forum, 2013).

Other services that are important to privacy are identified in the Privacy Management Reference Model and Methodologies (PMRM) committee standard (Sabo et al, 2013). Of note is that, the present PMRM version does not yet explicitly include the anonymization or pseudonymization privacy services, although they may be rolled up in its validation service. The 8 PMRM privacy services are agreement, security, usage, validation, certification, enforcement, interaction, and access. The privacy controls in NIST 800-53 Rev 4’s Appendix J may be easily mapped to many privacy services. Privacy services apply to multiple stakeholders in complex systems. As such, it will be strategic and responsible for software engineers to adopt a culture of auditable privacy protection enhanced by proposals such as ours and the OASIS PbD-SE TC’s support of pragmatic diagramming and documentation tools to show compliance to privacy regulations when dealing with linking big data and potentially sensitive sense-making and revelations.

IV. Prototype: Extending Visio with Privacy Services

![Prototype's Privacy Services Ribbon](Fig. 4.1)

We propose and create an MS Visio extension ribbon in Visual Studio as shown in Fig. 4.1. We called it a Privacy by Design (PbD) ribbon as it supports the software engineer to embed privacy into software in the early phases of development as per PbD principles. We also propose to add Anonymity and Pseudonymity services to the ribbon shown in Fig. 4.1.

The ribbon is automatically loaded at Visual Studio application startup. Software engineers can simply drag and drop the privacy services into their UML diagrams to illustrate at what points privacy services are to be introduced and used in their software applications. An advantage of an UML use case diagram is that it allows
people who understand the problem, and people who
design and implement the information technology
solutions, to communicate understanding of the various
use case scenarios in a complex system with multiple
stakeholders. For privacy requirements communication, it
is a useful visual aid in speeding up the privacy
requirements analysis phase in software engineering.

Importantly, problem domain experts and software
engineers, who share these diagrams, can specify the
different nuances, graduations, and granularity of service
required at each point of personally identifiable data flows
from one stakeholder to another. Raza et al. (2012) give
details of the choice of anonymization services, such as
semantically-swapped data, generalized data, or
suppressed data that software engineers can select for
privacy services implementation.

Details of our prototype use are given in the following
section.

V. UML USE CASE DIAGRAM’S PRIVACY
EXTENSIONS APPLIED TO BIG DATA APPLICATIONS

Most systems are too complex to be represented in one
page so software engineers utilize larger component use
cases to hide the system complexity and to handle scale in
the use case diagram. The use cases are composed of
many smaller use case scenarios. There is a similar
scaling problem when representing privacy requirements.
For instance, communication lines between a few actors
and use case scenarios will lead to many instances of
the same control with many communication lines/connections
between the various actors and components, rendering the
diagram unmanageable and difficult to understand. For
instance, if a few actors communicate with a few use case
scenarios and such communication requires SSL, the SSL
privacy control will appear on each communication line,
which will clutter the diagram. Clearly, some
representation is required to reduce the clutter and support
scaling of the diagram as the number of privacy service
operations that are required increases. We propose the
use of a super container as is shown in in Fig. 5.1. The
super container will host all the privacy services controls
required for a use case diagram and reduce the diagram’s
complexity by avoiding the creation of multiple instances
of a privacy service.

We now describe how the communication lines are
represented to connect the privacy service operations with
the actors and other controls, such as sub use cases. Fig.
5.2 shows a simple use case diagram wherein an actor
data scientist) is running a program that analyzes
historical patients’ treatment records without privacy
considerations being designed into the Watson-like
software system. The scenario we describe here is that
the data scientist tests the program on a sample of the
targeted big data and then submits the program for
execution to the system hosting the complete data set.

The scientist thus does not receive a whole copy of the
patients’ data, only the program’s output. Before the
treatment analysis application receives it, the patient data
should be pseudonymized while the output from the
program to be used or consumed by various actors may be
anonymized by degrees, or not at all, depending on
context. Before the output from the program is given to
the scientist, he/she needs to be shown a privacy notice
informing her/him to not to share or make the data in her
custody public and to not attempt to re-identify the output
data by linking it with other data sources. Furthermore,
the scientist must explicitly agree to comply with the
conditions stated in the notice.

The standard use case diagram of Fig. 5.2 does not
represent any of these privacy requirements. Furthermore,
UML use in other works has concentrated on modeling
access control at the class level; we are not aware of any
controls that address representation of the security and
privacy requirements in UML use case diagrams other
than those proposed in this paper. It is difficult and messy
to capture the privacy requirements in the use case
diagrams by using ad-hoc representation with special
annotations of privacy requirements. By using
components that are well-understood by modelers, and
directly hooked to privacy services, as proposed herein,
however, a business analyst would easily represent the
privacy requirements for the use case scenario as shown
in Fig. 5.3. The privacy super container is on the
communication line between the scientist and the sub use
case. It contains three privacy controls; (i) the first control
specifies the requirement of showing the privacy notice,
the use of output data by the program, and (ii) the scientist
and obtaining an agreement from her/him, (ii) the second
control specifies the requirement for pseudonymization of
the data before it is used by the program, and (iii) the
third control specifies that anonymization needs to be
applied on the data output from the program. As the
connection line in Fig. 5.3 from the data scientist is
connected to the super container, and also since the

Fig. 5.1 Privacy Services SuperContainer Control

Fig. 5.2 View Alternative Patient Treatments Use Case

communication line from the container is connected to the sub use case, all controls within the super container apply.

As there is only one actor, there is no confusion regarding to which actor the communication line, between the super container and the use case, belongs. If there is more than one actor, the communication lines need to be labeled as will be seen in the next example.

Fig. 5.4 shows a UML use case diagram without privacy controls. It has two actors, a data scientist, as in the previous example, and a doctor. It is the same scenario as in the previous case, with the addition of a doctor who needs to review recommended treatments. The doctor also needs to be presented with the privacy notice and the system also needs the doctor’s agreement to the conditions specified in the notice that may involve conditions from a patient’s consent directive. Data shown to the doctor needs to be pseudonymized. For this case scenario we also have the requirement that the system must communicate with the scientist and also the doctor over secure channels. How these privacy requirements are represented using the privacy controls is shown in Fig. 5.5. As in the previous case the data scientist is connected to the container, signifying that all privacy controls apply. But, as there is more than one actor, the connection line between the scientist and the container is labeled with “Ds”.

Similarly, the line between the container and the View alternative patients treatments sub use case is also labeled with “Ds”. This signifies that all privacy requirements, specified by privacy controls within the container, apply to the scientist and the View alternative treatments sub use case: communication must be over a secure channel, pseudonymization on input data must apply, privacy notice must be given and agreement obtained, and anonymization must be applied on output data. For the doctor, only three controls apply: communication over the secure channel, pseudonymization of data, and privacy notice and agreement. Anonymization is not applied. Consequently, the doctor actor is connected directly to the applicable privacy controls within the container. Furthermore, the doctor’s communication lines need to be labeled to properly identify the connections between the doctor actor, applicable privacy controls, and the doctor’s sub use case.

Our final example is a use case diagram that includes four actors and four sub use cases, as shown in Fig. 5.6 without the privacy controls. Fig. 5.7 shows the use case with our privacy controls. In it, the data scientist actor, as before,
concept of l-diversity, is applied as per Fig. 5.7. Another new actor is a head nurse who views a specific treatment record – only secure communication is required. The nurse actor is connected to the Security control and then to the View treatment sub use case. The doctor is now connected to two sub use cases.

As before, the doctor is connected to the Review Recommended Treatments sub use case, which requires pseudonymization, notice and agreement, and secure connection. He/she is also connected to the View treatment sub use case – in which case only a secure connection is required.

In summary, this section shows how UML can be usefully extended to embed privacy requirements in any Big Data and other software application.

VI. RELATED WORK

The field of engineering privacy is nascent particularly at the requirements specification phase. Ours is the only work in the literature at the intersection of privacy, Big Data, and UML for requirements analysis modeling. The most closely related work to ours is in privacy requirements engineering at Carnegie Mellon (Bijwe and Mead (2010), Abu-Nimeh and Mead (2009)). However, none of those works utilize the ubiquitous UML standard.

Hybridized computer scientists working in business and policy (Cranor and Spiekerman, 2009, Jutla and Bodorik, 2005, Jutla et al, 2006), and policy makers working with computer scientists (e.g. Cavoukian and Jonas 2012) have attempted over the years to bridge the communications gap between privacy policy makers, business domain experts, and software engineers in terms of privacy requirements engineering and privacy architectures. The communications challenge is not simple as software engineers can operate over an entire spectrum of engineering tasks from business requirements analysis to programming and unit testing. In between there are many tasks not limited to business process, architecture, and user interface design tasks, and privacy services and implementation selection tasks. Indeed, policy makers have a steep learning curve regarding understanding software engineering in depth. Existing tools are very specific to tasks and currently do not provide support to software engineers and architects to visually model privacy controls when developing a system. There are no

Fig. 5.6. Use case diagram with four actors

Fig. 5.7 Privacy controls for more actors

is connected to the privacy container signifying that all privacy controls within the container apply to the data scientist’s interaction with the View alternative patients treatments use case scenario. However, there is additional detail in that there are two anonymization methods specified within the Anonymization control.

Suppose now that the data scientist has requested and received full and extended access to an anonymized version of the big data set in order to troubleshoot a problem issue. As the scientist is connected to the container, and not directly to the control, the default method, k-anonymity with large k, is specified for her. The public researcher is a new actor that accesses data on which a strong anonymization method, based on the
suitable tools, apart from our proposal, available to incorporate a privacy service in UML use case diagrams while creating UML analysis models for a system.

Some large-scale efforts in the Health sector use UML class diagrams and use case templates to communicate important privacy processes. For example, the HL7 health standards group\(^3\) uses UML and customized extensions to explicate the consent directives management processes. While it does not provide a way of extending UML to support visualization of privacy services at the analysis or any other levels, the HL7 work provides insight for the complementary recursion and integration of both approaches. To be clear, our work on extending UML for use case diagram visualization fills a gap. It will allow the software engineer to design in and see the big privacy picture before getting in the weeds of the UML expansion that occurs when software engineers drill into details. The value of having our privacy extensions to UML is that the software engineer will understand all the interface points at which she needs to invoke specific privacy services, and hence will know that she needs to recursively drill down at those points!

Textual documentation can make learning formidable and hence visualization is a key for engineers to turn to tap into the vast knowledge of the domain expertise of the policy makers and problem domain experts. One work that supports visualization is the Privacy Policy Visual model (PPVM) (Ghazinour et al, 2009). The PPVM model represents each privacy policy component with different symbols: a data collector is represented by a house; entities and attributes are shown by rectangles and ovals; a relation is expressed by a line segment; the privacy policy is represented by a note symbol with predicates \(P:\) purpose, \(G:\) granularity, \(V:\) visibility, \(R:\) retention, \(C:\) constraint; group attributes are expressed by drawing circle around attributes; and default values are shown by text following a predicate. PPVM models and their symbols have not become standardized or suggested as UML extensions. However, it may be a useful tool to complement to our work as it provides a visual representation of a textual privacy policy which supports requirements solicitation, and can be an input to the stage of building UML analysis models such as use case diagrams. SPARCLE (Server Privacy Architecture and Capability Enablement) was developed to facilitate privacy policy creation, implementation, and compliance monitoring (Brodie, et al 2005). This tool helps users to create privacy policies, relationships, and access rules using natural language or well-defined (structured) format that are then translated into machine-readable format. None of SPARCLE’s tools allow a user to visually model the privacy policy rules.

Spiekerman and Cranor (2009) provide a binary classification of approaches to engineering privacy:

- privacy-by-policy that focuses on notice and choice principles, and privacy-by-architecture that focuses on pseudonymity, anonymity, and client-side protection architectures. Jutla and Bodorik (2005) propose a client-side socio-technical architecture, supporting multiple personas, and multiple service agents to protect user’s privacy. International start-up companies, e.g. Privowny.com, many hooked to the Vendor Relationship Management\(^4\) and the Personal Data Ecosystem umbrella projects, are now commercializing services that bear close resemblance to agents documented in (Jutla and Bodorik 2005, Jutla et al, 2006). Cranor (2002) was a major contributor to P3P, Platform for Privacy Preferences, an XML-based language for privacy to facilitate automation of privacy-related tasks. P3P attempted to help organizations efficiently and transparently communicate their privacy practices to their customers. Spiekerman and Cranor (2009) contribute a framework with characteristics for privacy-friendly system design. The framework is based on a spectrum; from the user as being identified to being fully anonymous. Our work is complementary and advancing as it directly facilitates the software engineer to build an UML analysis model of a system that embeds de-identification and other privacy services.

Another closely related work is the OASIS Privacy Management Reference Model and Methodology (PMRM) (Sabò et al, 2013). The PMRM shows the relationships among privacy regulations, policies, best practices, guidelines, controls, operational services, technical architectures, and implementation mechanisms for privacy. PMRM provides a step-by-step methodology to elicit and document comprehensive privacy requirements for any use case. Thus its use case output can be used as an input to our privacy-extended UML use case diagrams which form an important part of the analysis models for software systems.

VII. SUMMARY AND CONCLUSIONS

In 2010, Privacy by Design (PbD) (Cavoukian, 1995) became a unanimously acclaimed global privacy standard by the body of International Data Protection Commissioners. It seeks to influence technology design, business practices, and physical infrastructure by embedding privacy protection at its core. The PbD standard has tremendous influence on policy frameworks around the world. In 2012, the draft update of the European Data Protection legislation included adherence to Privacy by Design principles, and the US FTC released its final report on protecting consumers’ privacy with a recommendation that companies adopt Privacy by Design in building consumer privacy protection at every stage in their product or service development. Also in 2012, one of this paper’s co-authors convened an international standards organization OASIS Technical Committee

\(^3\) http://www.hl7.org

\(^4\) http://cyber.law.harvard.edu/research/projectvrm

(PhD-SE), called Privacy by Design for Software Engineers (PhD-SE), and led the development of its charter along with Dr. Ann Cavoukian. One of the PhD-SE OASIS TC’s core tasks is to map the 7 standardized Privacy by Design principles to UML so that software engineers can easily embed privacy requirements into their software.

This paper focuses on one part of the R&D work supporting Privacy by Design concepts. We illustrate how UML use case diagrams extended with privacy service components can help engineers to quickly embed privacy requirements into their analysis models. These high-level diagrams are popularly used to rapidly enable communication among software engineers, developers, problem domain experts, privacy officers, policy makers, and auditors. Similar privacy requirements and services can cascade through other UML diagrams using different user interface icons — topics for future research. However, the nut has been cracked and consultations with the software industry show that our approach has legs.

The pace of digitizing records, linking them to external repositories of knowledge, seeking similarities, patterns, and new breakthroughs impacting humans with personally identifiable information is rapid and transformative. The elements of public trust and organizational privacy compliance are far from settled as we balance them with societies’ and sciences’ need for more open data. Important communication tools, such as the privacy-enhanced UML proposed herein, will be essential to ensure secure and private systems and information flows. Further, these tools can help educate software engineers, system users, and other stakeholders around using privacy services to protect an individual’s freedom and rights to be left alone and forgotten.

Only through multi-stakeholder communication, collaboration, and mutual understanding can Big Data privacy issues be meaningfully addressed. Given the rapid birth and growth of Big Data platforms, it is urgent for stakeholders to cooperate to advance the quality of human life and economies while simultaneously fostering privacy protections to responsibly harness our new opportunities.

VIII. REFERENCES


Jutla, D. N. (2012b). Presentation to the Privacy Commissioner of Canada and her staff of self-authored reports on Privacy governance for software organizations and mobile app developers, and Personal data that mobile apps developers are monetizing and sharing with others, March 8, 2012.


