2. PROJECT DESCRIPTION:

2. PROJECT DESCRIPTION: A strange feature of the 21st century is that while there is *much we can learn* from each other, there is *little we dare to share*. How can we enable more effective data sharing?

The Internet lets anyone in the world transmit gigabytes of data from anywhere to anywhere else. Data miners can use that data to uncover significant patterns, even for large data sets [LDM09]. The same data mining tools can detect when new data is somehow strange and should be investigated or ignored [Wong05]. Otherwise, the models learned in this way can be incrementally improved as new data arrives [DL10]. Nevertheless, inter-organizational information sharing is inhibited by (a) privacy statutes; (b) regulations that limit the distribution of non-public personal information (NPPI); and (c) by organizational fears of disclosing confidential, sensitive, or proprietary information. For example*, in the field of medicine*: the Real-Time Outbreak Detection System (at U. Pitt.) seeks out-break patterns in data from healthcare providers. While that data is de-identified in accordance with HIPAA safe-harbor rules (removing 19 kinds of identifiers), privacy concerns make some participants hide information vital to tracking disease patterns; e.g. number of visits by ZIP code [Cli04]. For another example, *in the field of software engineering*: planning for software development efforts can be facilitated using patterns found in all projects. Boehm reports that the longer an error is left in software, the more expensive it is to remove [Boe76]. The size of these effects is still unclear: software developers usually do not publish their (e.g.) defect rates just in case this gives their competitors an unfair advantage in a competitive bidding situation.

Rather than cajole organizations to work against their own self-interest and expose their data, we need to *refine data mining techniques* that can *work in the real world*, by allowing data to *stay protected* while *building common knowledge*.

Our proposed work is very different to other research on privacy & data mining. Most of that research proposes *exposing all data* after anonymoization by (e.g.) adding random noise [Bec80,Agr00, Vai04]; or generalizing specific data [Swe02a, Mas07]. Sadly, that research *suffers from very limited verification*. Fung et al. report that one data set (the 48,842 records of ADULT; see http://goo.gl/jvxb) "is the de facto benchmark for testing anonymization algorithms" and list 13 papers that use it as *the only test case for their algorithms* [Fun08]. This limited verification is troubling. Brickell and Shmatikov report experiments where to achieve privacy using standard methods like k-anonymity and ℓ-diversity "requires almost complete destruction of the data-mining utility" [Bri08]. Their conclusions, based on just one data set (again, the ADULT data set) may not be externally valid. However, such disturbing results clearly demand more verification of privacy methods, ideally on more data sets from more sources.

Our approach, is based on three principles:

- *NO DATA EXPOSURE:* Based on decades of work in data mining and data sharing [Men03a,Men06, Men07a,Men09a,Mor09,Mor10,Shu99,Shu00a,Shu00b,Shu07], we assert that it is *very unlikely that organizations will expose their data.* However, some communities might form *trusted enclaves* of data providers which, under strictly controlled conditions, will grant limited access to other enclave members. For example, in our approach, if a data miner is dispatched from a sender to a receiver then before that data miner returns to sender, the receiver imposes their privacy restrictions to expunge conclusions they wish to keep private.
- *EXTENSIVE VERIFICATION:* To be convincing, privacy methods need to be assessed on records from multiple sources. Hence, we test our approach using (a) numerous data sets from the medical and software engineering domain; and (b) the privacy restrictions associated with those data sets.
- *DETERMINING THE PRICE OF PRIVACY*: We will selectively increase the privacy restrictions on our test data until our data miners stop working. In this way we will report the price of privacy; i.e. *how much can we protect out data* before losing the ability to *make useful conclusions*.

We argue that this proposal is very timely since, in the very near future, there will be much restructuring of federal privacy regulations. Already, there is much debate on the value (or otherwise) of the 1996 HIPA regulations [Ness07,Ness09]. Also, responses to electronic terrorism will require granting controlled government access to more data sets (see http://goo.gl/gvpr). *In order to best redesign privacy legislation, we need a better understanding of the cost-benefit trade offs associated with privacy and data mining efficacy.* Hence, this proposal.

Note that our emphasis on extensive verification using data collected from multiple organizations will be resource intensive and required collaboration from multiple sites. Hence, we propose this work as a LARGE project under the category of Trustworthy Computing.

2.1a Research goals and anticipated results: The goal of our research is to better understand *the cost-benefit trade-offs between increasing privacy constraints and decreasing data mining efficacy*. We specifically address three areas of important economic and social benefit: *software cost estimation*, *software inspection control*, and disease patterns in communities. These tasks are exemplars of a wide class of activities where groups are engaged in similar activities but cannot share data due to institutional or legislative or social considerations.

In order to achieve these goals we will address the following research tasks:

1. **Implement a privacy-based data mining without exposing private data.**
2. **Improve the state of the art in *software cost estimation*, *quality inspection control* and *recognizing patterns in chronic disease management*.**
    a) *Software cost models require calibration*. Rarely does an organization have enough data to calibrate their own models. The best models are built after sharing data from many sites.

    b) *Quality inspections can be too expensive to conduct on all parts of an engineering product.* Given sufficient experience, it is possible to prioritize what to inspect first, second, etc. However, rarely is there enough experience without one organization to learn the best prioritization.

    c) *Chronic disease management is often complicated by conditions that are infrequent at one site, but may occur at repeated sites*. If health organizations can share data from enough sites, then it may be possible to gain enough information to learn effective treatments.

We expect that our work will cover these three specific areas, will result in their improvement, and it will accelerate the broader adoption of our distributed data mining methods.

2.1b Intellectual Merit: Two unique features of the 21st century are the increasing use of electronic records and an increasing awareness of the security and privacy concerns about that data. Learning in such an environment is a complex task. While we seek models that work on as large a population as possible, we also need to recognize when locally learned lessons give different, and better, results than general conclusions. Hence, we need better ways to apply and compare the results of data mining from numerous results. Such comparisons are impossible unless some degree of access is permitted. However, such access is often blocked by security or privacy considerations.

The PIs have more than two decades of combined research experience in applying data mining techniques to solve real world problems, such as: defect prediction, effort and cost prediction, controlling the inspection process, defect location, optimization of non-linear models, etc [Gay10, Koc10, Men00, Men03a, Men03b, Men06, Men07a, Men07b, Men09a,Mor09,Mor10,Shu99,Shu00a, Shu00b, Shu07]. A repeated issue in all that work was obtaining access to data. For example, since 2006, we have tried to obtain permission to apply PI Menzies' data mining methods to PI Shull's software inspection data. The reply from our business partners has always been the same:

- The data cannot be leave the firewalls to travel to Menzies' group at WVU;
- But if the data miners could inside the local firewalls, and if the local business users could *audit and censor* the results before they are distributed, then that would be permissible.

Formally, these business partners are requesting (i) a *distributed data mining solution* where (ii) the learned models from the data miners are in *some human-readable* (and hence, *human-auditable*) format. Clearly, one major issue with such an architecture is that if business users can censor the data mining results, will we lose data mining efficacy? This research proposal was designed to address this question.

If we can show it is possible to build privacy-aware distributed data miners, and that the conclusions of those data miners are not unduly damaged by privacy restrictions, then this would usher in a new age of trust where data owners understand they can retain control of their data while still coordinating and sharing with other groups.

2.2 TEST DOMAINS: This research will impose increasingly onerous levels of privacy restrictions on a distributed data miner executing in two domains: (1) hospitals; (2) software engineering companies. At first glance, these domains may appear different. However, in terms of data sharing, they are very similar:

- Both need to share data while at the same time, *retaining privacy*.
- Both can be modeled as a *nested trusted enclave*; i.e. a tree of information sources in which parents can only see their children; and where all nodes strive to retain privacy.

The rest of this section describes these test domains.

2.2.a Sharing, Privacy and Learning in Software Engineering: Software is everywhere. Software has become a critical enabling technology for realization of functions central to our society. For example, most new cars are executing tens of millions of lines of software code, controlling everything from your brakes, to the intensity of the headlights, to the volume of the radio [Cha09]. Our energy generation, distribution systems; and even the pacemakers that control the beat of our hearts are relying on software. Hence, software failures can damage vital infrastructure. For example:

- In 2005, more than a decade after the opening of the Denver International Airport in 1995, and after spending $230 million on the troubled computerized baggage-handling, United Air Lines Inc. gave up on the failed project The troubled computerized baggage-handling system never work as designed and was responsible for a delay in opening the airport in the first place (see http://goo.gl/qUmW).
- The FBI uses software to investigate terrorism. A new Virtual Case File system was commissioned in 2001, but delivered until December 2004 Finally, in 2005 the FBI scraped the troubled $170 million computer initiative completely (see http://goo.gl/7u3A).

Clearly, software engineering (SE) as a discipline lags behind in terms of its ability to provide engineering processes which deliver artifacts of predictable quality within the required time frame. One approach to this problem is empirical software engineering which strives to find the patterns of success and the patterns of failures seen in real-world data from software projects. Advanced AI techniques such as data mining can find patterns in project data that predict for some quality variable such as number and/or location of bugs, development time, etc. Recent results from this work includes the following empirical patterns discoveries which find patterns:

- That reduce the effort of inspecting code by 71% [Tosun10];
- That reduce development effort, without incurring the penalty of greater defects [Men09b];
- To predict defect locations that are 1.5 to 3 times better than industrial practice [Men10b].

Sometimes, these empirical patterns apply only to a particular suite of software or an organization [Kit10]. However, recent results   suggest that some of these empirical patterns might even apply to multiple software suites or organizations [Tur09,Koc10]. That is, if software organizations dared to share data, then could use each other's data to (e.g.) manage new kinds of projects that had not been attempted locally, but which had been tried elsewhere.

There is a problem, however, with such data sharing. Extracting project data from organizations is very difficult due to the business sensitivity associated with the data. Recently, open source code repositories have become a rich source of software *product data*. However, software *process data* (e.g. what analyst capabilities lead to what development effort) is still very hard to obtain:

- Boehm (personnel communication) was able to collect 161 project records relating to development effort despite 30 year of work with many companies in the USA and China
- In our own work after two years we were only ever able to add 7 records to our NASA wide software cost metrics repository [Gre09b].

We diagnosis the problem as a *lack of trust*. Software organizations do not trust each other to share data, lest it gets used against them in (say) during competitive bidding. The goal of our distributed mining is to enable a consortium of companies to share data, without any of them revealing critical information.

2.2.b Sharing, Privacy and Learning in Hospital Medicine: This medical domain is concerned with the quality of patient care which is strongly related to the availability and quality of data. It is a domain that is ripe for data mining technology. The trend to consolidation in the healthcare industry with individual standalone community based facilities being acquired and integrated into regional healthcare systems provides an opportunity for localized data mining within the system. Administrators can look for patterns of best practices and errors across the individual departments, facilities, and practice groups. For example, at the Kimball Medical Center (an affiliate of the Saint Barnabas Healthcare System in New Jersey), they have devised a data mining process that found “golden nuggets” of clinical quality data that helped to improve patient care while providing best-practice information to physicians. [Vel08]

The ability to effectively share information, process it, and use the results in clinical decision support, while respecting patient privacy and ethical regulations in the entire process, can have significant impact on the quality of care offered. [Xia09]. Mining of existing records related to chronic conditions such as diabetes, obesity, and cardiovascular disease is expected to lead to more effective treatment and prevention [Rak10]. Diseases develop and evolve over time so modeling the treatment sequentially is also expected to be beneficial [Rak10].

It may not be sufficient to make public health conclusions based on data from a single site. Data pertaining to patient safety is collected not only by physicians but also by the government and by insurers. Mishaps, however, are still investigated episode by episode rather than through collective efforts. Such widespread collective efforts can yield significant resuts. In 1973, Wennberg and Gittlesohn [Wen73] reported significant variations in the rates of common surgical procedures across small geographic areas of Vermont that were not attributable to differences in the patient populations. The results have been confirmed across the US in subsequent studies ("the Dartmouth studies"). Further, these studies found that increased use of medical services has not been associated with improved outcomes, suggesting unwarranted costs. [Sut09]. Critics of the studies note the lack of sufficient data regarding other conditions and severity of illness [Zuc10]. For instance, additional testing may identify more diagnosed conditions for the same underlying illness [Son10]. According to Epstein [Eps10], the unexplained variations in care are really indicative of the lack of knowledge of best practices. Clearly, such regional variations motivate the wider sharing of information.

Although the necessity of collaborative sharing and learning has been recognized, there is little systematic knowledge sharing of clinical intervention outcomes [Xia09] [Gre06]. Privacy concerns are a mitigating factor impeding the sharing of data between entities. Competent health care depends on accurate and complete information. The collection and use of information relies on trust between the provider and the recipient and the belief by the provider that his privacy will not be compromised. The costs when the provider feels a lack of privacy include misdiagnosis or errors in care in the medical arena, false imprisonment or missed opportunities and severe monetary penalties in the private sector. The cost of a data breach for companies has risen to \$202 per lost record, up from \$197 in 2007. For the 47 companies audited in the study, those costs added up to \$6.6 million per incident [Gre09].

Recent legislation has required hospitals to express their records in a uniform manner. The Health Information Technology for Economic and Clinical Health (HITECH) Act is intended to spur the electronic exchange of health information to improve quality and improvement of care and reduce errors and unnecessary procedures while strengthening privacy and security of identifiable health information. (see http://bit.ly/gk3BO). A key outcome of this initiative is the mandated move to the use of International Classification of Diseases v.10 (ICD-10) coding scheme which provides finer granularity of coding.

This uniform data structure means that distributed data mining might be successfully applied in hospitals. Once the ontology of that site is known (e.g. if it uses ICD-10), then designing that agent to interface with the data it might find becomes a simpler engineering task. Formally, this means that data miners can reason about *horizontally partitioned data* (same tables, with rows stored at different sites).

Further, in our discussions with hospital administrators, it has become clear that if we give those administrators the same *audit-and-censor* functionality requested by the software engineering managers, then that would increase the willingness for receiver sites to accept and execute someone else's data miner (since they could audit and censor any out-going results, before anyone else sees them).

2.2c Trusted Nested Enclaves: Our thesis is that hospitals and software engineering companies can both be modeled as a *nested trusted enclave*; i.e. a tree of information sources in which parents can only see their children; and where all nodes strive to retain privacy. Such enclaves form are acyclic network where data miners of some parent node (at level N) can only access the data of their child nodes (at level N+1). Each child node also contains data miners that reflect on grandchild nodes (recursively).

It is simple enough to demonstrate the nested structure of information sources in hospitals and software engineering organizations. Figure 1 details a possible recursive refinement into several levels for the Software Engineering and medical domains. In general, such refinement levels will help align implementing our enclave learners with any hierarchically structure in other organizations and domain, such as finances, intelligent services, or even universities.

Formally, we say that enclaves are sets of *semi-honest adversaries* collaborating to try and learn from each other, without revealing too much about themselves (a semi-honest party follows the rules of the protocol using its correct input, but is free to later use what it sees during execution of the protocol to compromise security). In our work, we explore data sharing in trusted enclaves (e.g., see [Mor09, Mor10]). A trusted enclave runs agents that offer Continuous Compliance Assurance (CCA), where "compliance" is assessed with respect to some query describing the kinds of information that must not be revealed from the enclave.

Enclave level	Software Engineering Domain	Medical Domain

1	Research network (e.g., International Software Engineering Research Network – ISERN Empirical Software Engineering group – ESE, ISERN: http://isern.iese.de)	State (e.g., WV, MD, DC)
2	Research group (e.g. , FC-MD, SEI, IESE: http://www.iese.fraunhofer.de/index.jsp.)	Hospital Groups
3	Application Context (e.g., aerospace, education, financial)	Single hospital / unit
4	Projects/Programs (e.g., NASA, Keymind, DoD)	Departments (e.g., ICU, pediatrics, radiology,…)
5	Sub-groups (e.g., according to location, code packages, application context)	Private practice / remote clinics / physician specialties

Figure 1: *Refinement example of research enclaves for the Software Engineering and medical domains.*

Based on our work with government institutions, we assume the following ontology for our compliance restrictions: they are combinations of "and, or, not" around attribute range queries; e.g*., Age<21 or organization="nsf".* For example, the xpath query of Figure 2 shows compliance restrictions in that ontology. A DOD secrecy rule is shown that demands that the weight of its consignment cannot be revealed (heavy DOD consignments may be nuclear materials with heavy shielding). Note that our ontology assumption it is hardly controversial: we can find in the literature similar assumptions about privacy restrictions [Agr03]. However, as discussed below, it is an important assumption since when we come to selecting data mining technology.

```
// U.S. Privacy Act of 1974
not(boolean(//RAAR/CrewDiscrepancies/Cre
wDiscrepancy/CrewPersons/CrewPerson
[CitizenshipCode = "US" and
(boolean(SID) or boolean(DateOfBirth) or
boolean(PlaceOfBirth)or
boolean(Height)or boolean(Weight) or
boolean(HairColor) or boolean(EyeColor)
or boolean(DistinguishingMarks) or
```

Figure 2: *Top: a privacy restriction. Bottom: a secrecy requirement.*

We say that each node is owned and operated by the local administrators. One way to perform privacy preserving data mining in an enclave is via a data miner passed around a ring of enclave nodes. At the end of the ring, the data miner returns to where it started from to report its final conclusions. As it passes over nodes in the ring, the locals can restrain the data mining:

- *HIDING THE DATA:* The locals can run their own local version of the data miner to identify which parts of the data lead to rules they do not wish to disclose. These data sections can be hidden from the data mining being passed around the enclave.

- *PRUNING THE RULES:* Before a data miner leaves a node, the locals can prune any parts of the learned rules that overlap with the compliance assurance (e.g. Figure2).

Techniques for data pruning and rule pruning are discussed later in this proposal.

2.3 ALTERNATIVES TO ENCLAVES: Before going any further, we need to explain our preference for enclave-based data mining. Accordingly, this section discusses (a) alternative approaches to privacy and data mining that *do no*t use enclaves; and (b) why we elect not to use them.

Data mining is the process of finding patterns in data. In terms of privacy, the core problem with data miners is that, traditionally, they assume access to a global database of all the information. This is worrisome if the data miner includes in their learned model some pattern that should remain confidential. Fung et. al. [Fun10] distinguish two classes of research in this area:

- privacy-preserving data publishing (PPDP) and
- enclave methods that offer query control on multilevel secure databases.

The difference between PPDP and enclaves is as follows:

- With enclaves, the data remains private and but the enclaves publish their queries[1].

[1] Brickell and Shmatikov discuss an extension of enclaves, which we do not explore, where all nodes can access a global data dictionary, but not the attributes used in a query [Bri09]. Such an extension is not appropriate for our purposes since our users want to browse ranges to check that they do not violated compliance assurances.

- With PPDP, the data is published and any inference on that data remains private.

Research in *statistical databases* implements PPDP by allowing statistical information (sum, count, average, maximum, minimum, th-percentile, etc) to be accessible, without revealing sensitive individual information. PPDP techniques include*, query restriction, data perturbation*, and *anonymization*. The *query restriction* methods includes restricting the size of query results [Den79], controlling overlap in successive queries [Dob79], keeping audit trails of answered queries (to check for possible compromises) [Chi82], avoiding data cells of small size [Cox80], and clustering entities into mutually exclusive atomic populations [Yu77].

As to *data perturbation*, techniques include swapping values between records [Den82], replacing the original database by a sample from the same distribution [Rei84], sampling the result of a query [Den82], and adding noise to the values in the database [War65] or the results of a query [Bec80]. The effectiveness of perturbation is an open issue. Adding noise hides the details of individuals, but can confuse learners that (say) try to find the best place to discretize numeric data [Vai04]. Agrawal and Srikant [Agr00] offer one solution where Bayes' rules is used to reconstruct the original distributions using knowledge of the distributions used to add the noise. The problem with this technique is that the reconstruction gives us information about the original data values, thus violating privacy [Zhang07].

Regarding *anonymization*, a naïve approach is *trivial sanitization*; i.e., the removal of all identifying information from the data release. The problem with sanitization is that pinpointing exactly what constitutes identification information is difficult. Repeated patterns in databases allows for the simple identification of individuals, even after sanitization. For example, Sweeney notes that 87% of the population of the United States can be identified by linking the following attributes: gender, date of birth, and 5-digit zip code. In one dramatic example of this, recently the medical records of the Governor of Massachusetts was recognized, even in supposedly anonymous sanitized data set [Swe02b].

One widely-used anonymization technology is *k-anonymity* [Swe02a], which is based on the notion of a Quasi Identifier (qid) is a set of attributes that could potentially identify record owners. Under the k-anonymity requirement if one record in the table has some value qid, at least k−1 other records also have the value qid. A data set can be made k-anonymous by (say) generalizing the attributes; e.g., replacing all day/month/year records with just month/year. K-anonymity does not ensure privacy in the case of attackers using background knowledge on the groups returned by a query. For example, suppose the qid is <profession,gender,age> and it is found that 80% of female dancers aged 30 are HIV positive. If an attacker knows that that this data comes from a dance company, and that Emily (who is aged 30) dances there, they the attacker could infer with 80% probability that Emily is HIV positive.

Improvements to k-anonymity include Machanavajjhala et al.'s principle of *ℓ-diversity* [Mas07]. ℓ-diversity requires every qid group to contain at least ℓ "well-represented" sensitive values (at least ℓ distinct values for the sensitive attribute in each qid group). ℓ-diversity has the limitation of implicitly assuming that each sensitive attribute takes values uniformly over its domain; i.e., that the frequencies of the various values of a confidential attribute are similar. When this is not the case, achieving ℓ-diversity may cause a large data utility loss.

The efficacy of all these methods is open to question. Brickell and Shmatikov [Bri08] report simplistic trivial sanitization provides equivalent utility and better privacy results than supposedly better method such as k-anonymity or ℓ-diversity. As discussed above, their results are hardly conclusive (since they are based on a single data set). However, given all the problems discussed above with PPDP, we agree with Vaidya and Clifton [Vai04] that rather that struggle to secure a public data set, it might be better not to build that data set in the first place. Hence, for the rest of this proposal, we discuss distributed data mining methods over trusted enclaves.

2.4 DISTRIBUTED DATA MINING OVER TRUSTED ENCLAVES: In our enclaves, parents can see their children but not their grandchildren. How can a parent gain insight into grandchild nodes? To address this problem of recursive insight, we distinguish between *data mining* and *rule fusion*.

TOP-DOWN DATA MINING: Parents collect statistics from their children, while maintaining the privacy of each child. One way to achieve this is secure multipart computation (SMC). As described by [Vai04], SMC is a conversation between *N* parties, none of which want to display their data to another. For a simple example of SMC, consider the "no collusion" case where a parent node in an enclave wants to sum some value *k* across its *N* children. A random number *R* is passed to any child at random. That child adds its local *k* value, passes it to another child, selected at random. When all children are visited, the parent receives back the sum, removes *R,* thus accessing the actual sum. Note that no child can infer what are the actual values of *k* in the other children since those values are masked by *R*. Two open issues with SMC are *collusion* and the *runtime cost*:

- *Collusion* between children can make SMC computation insecure. If children *z-1* and *z+1* compare the values of the running sum, they can compute the exact value for *k* in child *z*. Assuming more than three children, we can fix this problem by passing the sum around in random order amongst the children (so child *z+1* never knows who was child *z- 1*).
- As to *runtime cost*, Vaidya and Clifton [Vai04] report that SMC can be remarkably slow. In one test case using SMC on a multi-node network that required some joins between different tables, it took 29 hours to build a 408-node decision tree from 1,728 examples. Clearly, SMC is not a recommended when nodes in a network must engaged in high bandwidth communication to achieve some results. Clearly, private top-down data mining support collusion avoidance and resolve the runtime cost.

BOTTOM-UP RULE FUSION: After a parent data mines their children, they broadcast the rules upward (after first deleting any rules that violate compliance). Grandparent nodes collect and fuse the rules from the parents. In this way, insights gained deep in the enclave can bubble upwards (but only if they are supported by multiple children). Once a parent builds a rule using fusion of child rules, then this new rules needs to be checked across the children. A rule's value can be determined by testing the rule on the children. If a rule's true negative, false negative, false positive, and true positive rates are a,b,c,d then that rule's *precision*= d/(c+d); *accuracy*=(a+d)/(a+b+c+d); *recall*=d/(b+d); *false alarm rate* = c/(a+c) and *"f" measure*= 2*prec*pd/(pd+pf). If a parent collects these a,b,c,d values by passing a rule over its children using SMC, then the value of a fused rule can be computed without violating privacy.

From the above, we can deduce seven essential aspects of the design of our distributed learners:

1) *RULE FUSION:* There must be some way to combine rules from multiple sources.
2) *COLLUSION AVOIDANCE:* In order to avoid collusion during SMC, each enclave must include a transaction manager whose task is to pass a computation around a ring of nodes in a random order.

3) *SMC OPTIMIZATION:* In order to support SMC, when a data miner is initialized at the start of a query, its internal frequency counts are distorted by a random amount known only to the creator of the miner.

4) *BATCH PROCESSING:* To avoid the computational overhead of unconstrained SMC, our data miners must be not dispatch thousands of queries across an enclave. Rather, they should be one-pass learners that can make their conclusion after a single round-robin traversal of a set of nodes.

5) *AUDITABILITY:* In order to let the locals recognize a compliance violation, then the output of the learner should match the ontology for our compliance described above. Hence, whatever learner we use, it should user high-level rules and not some arcane incomprehensible internal format.

6) *RULE PRUNING:* In order to let the locals censor rules that violate compliance, then the learner's model should contain parts, any one of which can be deleted.

7) *DATA HIDING:* If the locals are to remove the data that lead to rules that violate compliance, it must be possible to track backwards from any rule to the data that generated it. This data should then be hidden from any incoming SMC requests.

Of the above criteria, items (3) and (5) lets us quickly rule out many data mining technologies. Those technologies include (a) discrete learners that find either association rules that report frequent patterns of attribute ranges that occur together [Agr91], or classification rules/decision trees that find frequent patterns between independent attributes and one dependent "class" attribute [Bri84,Qui92]; or (b) probabilistic methods such as fuzzy learners or Bayes classifiers [Wit05] or the EM clustering algorithm [Dem77] that represent different classes as distributions; or (c) methods that use probability distribution propagation over a directed graph like Bayes nets, neural nets [Hin92], distributed Kalman filters [Olf07], or non-parametric belief propagation (NBP) [Tse04]; or (d) instance-based learners that reason about examples nearest some test instance [Aha91].In order to support *AUDITABILITY*, the ontology of the learnt model must match the compliance restriction. Hence, instance-based learners, which generate no model, are inappropriate for our work.

The discrete learners seem most suited to our task since the and-or-not nature of discrete rule conditions are closest to the ontology of our compliance restrictions. However, many discrete learners are not suitable. Decision tree learners like C4.5 [Qui92] recursively divide the data set and call themselves on each subset of the data. This means repeated inspection of subsections of the data- which fails the *SMC OPTIMIZATION* criteria. Recall from the above that using SMC took 29 hours to build a 408-node decision tree. Similar issues exist with association rule learners like APRORI [Arg91]. This algorithm finds frequent item sets of increasing size and, for each such larger set, it conducts a repeated search of the data to count the occurrences of that set.

At first glance, the probabilistic models are inappropriate since our users require categorical rejection rather than some partial probabilistic pruning. This is unfortunate since probabilistic methods such as Bayes classifiers have some of the properties that we desire such as one-pass incremental learning. However, such classifiers build a single model of the data expressed in a format that is quite alien to the compliance ontology (distribution information on each attribute, in isolation from all other attributes, divided into one distribution for each class).

Recently, we have had success with rule generation from Bayes classifiers [Cla05,Gay10,Mil08]. TAR5 grows sets of interesting ranges (given discretized data, there exists one range for every attribute=value pair). The ranges are sorted on

Round0	**Round1**
Top of stack	**Top of stack**
78 if sex=female	78 if sex=female
71 if class=1st	74 if class=1st and sex=female
68 if age= child	71 if class=2nd and sex=female
65 if class=2nd	72 if class=1st
	68 if age=child and class=1st
	68 if age=child
	68 if age=child and sex=female
	65 if class=2nd

Figure 3: *Rules found by TAR5. Left-hand-side numbers are accuracies predicting for survival from the Titanic. TAR5 sorting ranges from the last round, combining the better ones (selected stochastically, favoring those nearer top of stack), then scoring and sorting the new combinations into a new stack.*

to a stack according to how well they selected for a preferred class (see Figure 3[2]). TAR5 combines ranges at random to form rules (favoring ranges that appeared higher in the stack). TAR5 runs one stack per classification. Each stack finds a rule set that selects for classification, and avoids the others.

The stack is initialized by passing all ranges through the scoring scheme of Figure 4. TAR5 repeatedly selects *R=2* items from stack (favoring items with higher scores). Each selection is combined into a conjunction, scored, and sorted back into the stack. If it scores worse than existing items, it sorts lower on the stack (becoming less likely to be used in future). Otherwise, the new item moves up the stack, making it available for future selects. As TAR5 runs, items can grow in size as more useful conjunctions are discovered and combined (Figure 4 shows TAR5's rule growth using data on who survived the loss of the Titanic). TAR5 terminates when the score of the rule on top-of-stack stabilizes, at which point, TAR5 returns the top item as the best selector for some class.

When applied to the task of defect prediction for software modules, TAR5 out-performed standard learners such as Naive Bayes or decision-tree learners [Mil08]. It has been used at NASA to tune the settings of complex guidance, navigation and control flight systems [Gundy08, Gundy09]. In comparisons with state-of-the-art optimizers (a Quasi-Newton method that incremental updates a Hessian approximation), our method ran 40 times faster, and found better solutions [Gay10].

TAR5 was an experiment with one-pass learning. 80% of the algorithm's runtime arise from the repeated checking of the rules against examples of data. The algorithm removes that runtime by replacing that check with Bayesian evaluation heuristic. After one pass of the data, the algorithm computes just enough information to allow for the fast ranking of different rules without needing to pass again through the examples. In practice, the algorithm ran two orders of magnitude faster than earlier versions, and used an order of magnitude less memory [Cla05]. Further, as described in Figure 4, this Bayesian heuristic has proven to be remarkably accurate.

TAR5 sorts all ranges into its stack as follows. A table of observations containing N observations has labels $L_1, L_2, \ldots$ appearing in $N_1, N_2, \ldots$ examples (so $N = \sum_i N_i$). For any label L_i, we say Rest is all the other labels (i.e., $L_i \notin Rest$). If a range r appears at frequency $f(r)$ in all examples, and $f(r \mid L_i)$ in the rows labeled L_i, then r appears outside of the L_i rows at frequency $f(r \mid Rest) = f(r) - f(r \mid L_i)$. The likelihood of r being in label L_i, or in $Rest$, is $\frac{f(r \mid L_i)}{N_i} * \frac{N_i}{N}$ or $\frac{f(r \mid Rest)}{} * \frac{(N - N_i)}{}$, respectively. According to Bayes' theorem, the probability $P(L_i \mid Rest)$ that r

[2] Since we cannot show real data from the confidential databases of our clients, we must resort to examples based on publically available information. Hence, the example of Figure 3 is based on survival data from the S.S. Titanic.

Figure 4: *TAR5's Bayesian scoring method.*

2.5. PROPOSED APPROACH: While a promising start, TAR5 is unsuitable for data mining in trusted nested enclaves. A full solution to our distributed data mining problem must address the challenges of:

1) DATA COLLECTION;

2) *TECHNOLOGY DEVELOPMENT:*

   a. rule fusion,

   b. collusion avoidance,

   c. SMC support,

   d. batch processing,

   e. auditability,

   f. rule pruning,

   g. data hiding

3) *ASSESSMENT:*

   a. efficacy of the data mining,

   b. anomaly detection,

   c. model revision,

   d. and finally, assessing of the price of privacy

In this work, we propose extending TAR5 to address these challenges. The new system, called TAR6, will be tested on data and compliance requirements collected from hospitals and software engineering companies. Using that information, we will observe the effects of increasing or decreasing those compliance requirements on the efficacy of TAR6's rules (this last study will be used to better understand the trade-offs between data mining and privacy).

The first challenge is to collect the data required for this work:

Challenge 1: From multiple sites, collect real world data & their compliance requirements.

The introduction of this proposal lamented the poor state of the art in verification of privacy algorithms. Many papers assess their work on theoretical grounds or using a single data set. Clearly, this is not an ideal verification method. Real world data is notoriously quirky (what is true in one data set may be irrelevant in another [Men06, Men07d]. Privacy tool needs to be tested on dozens of real-world examples, ideally from multiple organizations.

The next few challenges relate to technology development.

Challenge 2: Add rule fusion to our learner.

Fusion means combining ranges from different rules from different sources, then evaluating the result. In the context of TAR6, a grandparent node in the enclave can fuse rules as follows. If P parents offer N rules (learned from their children), then there are N*P rules to fuse. Each rule, in isolation, will have some score (from Figure 4). The grandparent could sort the rules on that score and try stochastic combinations of those N*P rules, favoring those with higher scores. This is almost the TAR5 rule growth procedure described above, with one small variation. When a combined rule is scored, the grandparent sends the new rule back to each parent and asks them to score it from their children. Any combined rule that scores well will float to the top of the grandparent stack. Similarly, rules that worked well at one parent, but not on all, will float to the bottom.

Note that this fusion strategy is analogous to gossip networks [Dim06] where nodes compute some joint value using randomly sampled neighbors. However, where as gossip networks assume continuous distributions, TAR6 assumes that the knowledge to be combined is discrete rules. Also, note that TAR6's fusion strategy is much simpler than standard fusion methods. Much research has explored fusing the results from multiple learners [Toh04]. In the basic ensemble method (BEM), learners are run on various subsets of the available data. These learners use some features to predict for classes to find a function that returns the probability of the target classes. BEM returns the mean probability of items in the ensemble while the linear generalized ensemble method (GEM) returns a weighted sum of the conclusions of each learner in the ensemble (so, with GEM, the ensemble item that perform worst, contributes least to the overall conclusions). For some data sets, the combination rule is non-linear and complex. For example, Toh et.al. [Toh04]'s variant of GEM uses a Jacobian matrix for with different coefficients for each feature and target class. These coefficients are learned via multivariate polynomial regression. In our experiments, when we tried to scale fusion methods to hundreds of learners, computing GEM was too slow and the results little better than noise. Based on those experiments, we elect instead to explore rule fusion in the context of TAR6.

Rule fusion enables combining rules learned in multiple parents. This, in turn, requires that some rules exist in the first place. In order to use SMC to sample child data to build parent rules, we must:

Challenge3: add collusion avoidance

This is a systems engineering task. Some special node in the enclave will be declared "the manager". If some client node wants to poll N others, then it sends that list of nodes to the transaction manager. This manager orchestrates the SMC queries across that set of nodes such that no visited node knows who was visited before or after. Finally, the manager returns the results of the SMC computation. The client knows that the results come from certain other nodes, but not which particular nodes offered which particular data items.

The high cost of SMC was noted above. Our computations must avoid too many low-level queries.

Challenge 4: SMC optimization

Recalling Figure 4, we already have much support for believing that our learning can be done in one-pass of the enclave members. Hence, in theory, we can avoid the high computational cost of standard SMC. Of course, such theoretical beliefs must be tested on real-world data.

The next challenge addresses two concerns. First, in order to detect rules that violate privacy concerns, the ontology of the compliance regulation must match the generated model. Second, if some rule violates compliance, it should be possible to delete it without harming the rest of the learned model:

Challenge 5: auditability and rule pruning

In this respect, the current implementation of TAR5 would suffice. Recalling Figue 3, our rules already match the ontology of Figure 2. Secondly, our TAR algorithms contain a stack of separate rules, all struggling to extend themselves in order to float towards top of stack. Extracting one rule from this stack will not harm the processing of the remaining rules.

Challenge 6: data pruning;

Recall from the above that data pruning is the process of hiding data from an incoming SMC request such that it is difficult/impossible for that request to learn a pattern that the locals wish to keep private.

To meet this challenge, TAR6 will augment TAR5's rule generation process with a dependency graph of what rules lead to other rules, Before the locals at any enclave accept any SMC requests, they could run TAR6 to generate, e.g. the graph of Figure 5. Using this graph, the locals can rapidly select what data to hide from any incoming SMC requests. Suppose the locals wanted to hide some patterns or boast about others. For example, suppose the owners of the Titanic might want to (a) hide the information that second-class women receive nearly the same preferential treatment as first-class women; while (b) advertise that all female children have a better-than-most chance of survival. Queries to the dependency graph can select rows that lead to some observations, but not others:

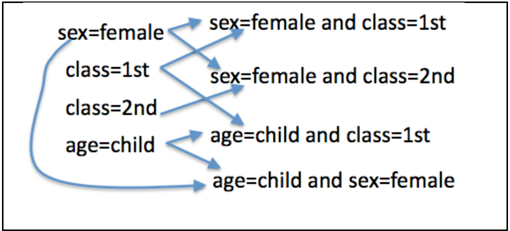

***Figure 5**: connections of smaller rules to larger rules from Figure 4.*

- Take each row of the data and present it to the left-hand-side of the dependency graph.
- If any range in the row is found in the root of the graph, then mark that row as *suspect*.
- Move the suspects right across the dependency graph, awarding points to suspects that match to preferred rules and subtracting points to rows that match to undesired rules.
- Hide the rows with H% lowest points.

Note that this algorithm is tunable: increasing H hides more detail while setting H=0 exposes all data to any incoming data miner.

Once the above challenges are meet, we would have a working distributed data miner over nested enclaves. With that in hand, we turn to the remaining challenges.

Challenge 7: efficacy of the data mining

These whole proposal assumes that our data miners can find interesting patterns in data from software engineering and medical companies. Using historical data of known past patterns, this assumption should be tested.

Challenge 8: anomaly detection

Once we have a working data miner, it would be possible for local administrators to test if rules learned elsewhere are working as expected at their local site. For example, in the medical domain, the locals would suspect an anomaly if a rule predicting for low post-operative infections (that is published by many other nodes) does not work well at this local site. Such anomalies could be an early warning sign for some previously undetected change in local conditions.

Challenge 9: model revision

In all the domains we study here, it is unlikely that once a model is learned, that it will remain constant for all time there after. For example:

- In software engineering domains, new technologies are constantly appearing.
- In medicine, health patterns change along with the seasons or as new drugs /diseases appear.

Hence, just as important as learning an effective model is knowing when to *change* an existing model. Our proposed model revision operator for TAR6 is as follows.

Recall that TAR5 executes by selecting and combining $R = 2$ items from its internal stack. TAR6 modifies this selection policy as follows: *select* $R \geq 1$ *items from the stack.* There are two interesting cases for this selection policy:

- If R > 1 then TAR6 is trying to combine existing items on the stack into larger conjunctions. That is, when R > 1, TAR6 is *building bigger rules*.
- Also, if R = 1, TAR6 is selecting one existing item from the stack, and restoring it. That is, when R=1, TAR6 is *reviewing old rules* (and possibly discarding them).

During this second case (review of old rule), it is possible to update TAR6's rules if changes to the observations have changed. If the Figure 4 scoring for a rule has altered, then TAR6 can demote (or promote) a rule according to its effectiveness on the latest observations.

Challenge 10: assessing the price of privacy

With all the above machinery in place, we will be able to perform distributed data miner where the results of that mining are audited and censored by compliance assurance requirements. The major question of this research can now be addressed: how much does compliance assurance hinder data mining? To test this, we will apply domain modeling to our software engineering and medical domains to determine appropriate ways to strengthen and/or weaken the compliance assurance requirements. The effects on the efficacy of the data mining will be noted and we will offer conclusions about how much privacy requirements effect data mining efficacy.

2.6. RESEARCH PLAN:

2.7. EVALUATION CRITERIA.

2.8. EDUCATION/ BROADER IMPACT:

Benefits to society at large: Our specific research goals focus on issues of tremendous economic or social importance (better control of software projects; better understanding of disease patterns in society). More generally***,*** if we can show it is possible to build privacy-aware distributed data miners, and that the conclusions of those data miners are not unduly damaged by privacy restrictions, then this would usher in a new age of trust where data owners understand they can retain control of their data while still coordinating and sharing with other groups.

How will individuals at underserved institutions benefit from this grant? By funding this work at WVU, NSF will be promoting higher education in a region which, currently, lags far behind the rest of the country in terms of its population starting, or completing, a University degree. Appalachia, which includes West Virginia, remains one of the most economically depressed regions in the United States. This economic condition greatly impacts the number of young people choosing to attend college. In the 1990's, the gap between Appalachia and the rest of the U.S. in the percent of the adult population who are college graduates increased from only 6.1% to 6.6% [55]. In addition, the number of students seeking science and engineering degrees lags national averages: West Virginia and the rest of Appalachia are in the bottom quartile in the percent of science and engineering degrees awarded [161].

Also, by funding this work at Fraunhofer Institute at the University of Maryland, NSF will be promoting higher female involvement in engineering. The Fraunhofer has a strong track record of including women in software engineering research projects. 40% of the Fraunhofer technical staff are women, far above the latest numbers for women's representation in U.S. computer science degree programs [37]. Fraunhofer's support of women in science is particularly important since, at this time, the rates of increase in computer science by men is far greater than in women [37].

How well qualified is the proposer (individual or team) to conduct the project? The PIs on this project have the necessary experience to succeed on this project. We have extensive experience with empirical software engineering [6, 16, 17, 22, 44, 107, 108, 109, 110, 112, 114, 144, 145, 146, 147] including running large collaborative experiments [109, 147] and organizing repositories of software engineering data (Menzies has developed an open-source toolkit for the analysis of software engineering data [5, 48]. We are uniquely placed to (a) understand the most that can be learned from this data and (b) detect when learning is damaged by increased privacy restrictions. We have worked extensively on real work applications on data mining for software engineering [Men03a, Men06, Men07a, Men09a, Shu99, Shu00a, Shu00b, Shu07] and with data sharing amongst public hospitals and other government institutions [Mor09,Mor10]. That is, we are knowledgeable in the baseline results available from this data (when no secrecy restrictions are applied). With our contacts, we can find industrial and government data, plus their associated security and privacy restrictions.

How will this research be integrated into teaching ? Much of the research in this project will also be integrated into a classroom environment. PI Menzies teaches graduate data mining and all the tools will be used in that subject. PI Menzies also places all of his teaching materials on the web which means that any other data mining lecturer will be able to access tutorials, assignments, and lectures.

Dissemination of knowledge: We will make the developed tools and underlying technology for setting up enclaves freely available as open source system. Also, the PIs on this grant frequently publish in numerous research forums (IEEE TSE, IEEE Computer, IEEE Software, ASE, ICSE, etc). We hence anticipate numerous publications from this work.

Also, we have a goal of making some data from this work freely available for other researchers. The authors have an exemplary reputation of placing their data on-line (Shull ran the CeBASE repository [16] while Menzies is the webmaster of the PROMISE repositories: http://promisedata.org). However, with regard to the data studied in this work, the practicality of this goal will be assessed with respect to the business or federal government restrictions on the data.

2.9 RESULTS FROM PRIOR NSF WORK

Dr. Forrest Shull was a co-PI of: NSF Science of Design collaborative grant CCF0438933 and CCF0438923, $1M from 2/1/05 to 1/31/09 “Collaborative Research: Flexible High Quality Design for Software”. This project applied an empirical approach to investigate effective indicators for assessing the flexibility of software architecture, as well as assessments of V&V techniques which aim at improving flexibility under various conditions. The work produced a number of laboratory packages used to replicate studies and analyses at various collaborating sites. This support provided support for 9 graduate students over its existence. It produced 7 journal publications [15, 19, 53, 95, 96, 148, 167] 19 conference publications [2, 3, 4, 18, 29, 53, 60, 75, 83, 84, 125, 127, 128, 137, 139, 141, 168, 173] 2 Master theses.

Dr. Menzies was awarded in July 2008 an NSF grant (CCF-0810879, $350,000, end date June 30 2011) on “Automatic Quality Assessment: Exploiting Knowledge of the Business Case”, together with Dr. Bojan Cukic (Co-PI) from WVU. The task of this research is to explore the inner loop of software data miners improves the learning of defect predictors. Prior attempts at improving defect predictors have produced very little improvement, despite years of effort. This work produced a number of tools including the OURMINE package that is the basis for this work. This project has only just entered its third year but already has produced 3 journal papers [67, 120, 159] and 12 conference papers [35, 48, 52, 66, 67, 68, 72, 73, 105, 109, 129, 171]. Also, it has fully supported to completion one female Ph.D., one female masters and four others masters students to completion. It is also currently supporting one Ph.D. and one female masters student.

On August 15, 2010, Menzies starts a new NSF grant, CCF-1017330: “Better Comprehension of Software Engineering Data” ($500,000; a new grant with Andrian Marcus at Wayne State).

Research, first year:

To study the impact of protocol rules on the findings in an enclave, we will conduct a series of case-studies. For the case-studies we will use data-sets in the Software Engineering domain. Initially, we will use data-sets from real industry projects in the aerospace context to which FC-MD has access. If needed, additional data-sets from other application contexts, such as research environments or other industrial contexts, in the Software Engineering domain, may be added later on. For all case-studies, the trusted enclave will be defined as the FC-MD environment with different nodes depending o the case study. One such set-up, could, for instance, be the different (aerospace) projects which initially serve as a natural discrimination into different leave nodes of the enclave.

But before we start such a multiple project study, we will start with some smaller case studies focusing on data of single projects. These studies will repeat some analysis we already conducted on the data sets, so we know the possible results and can compare how privacy rules could affect them. The NASA inspection data which we already analyzed regarding influencing factors of the efficiently of inspections [Sea08] will serve for one of these initial studies. We will use the inspection data collected in different NASA Centers as a basis. The participation into nodes / PIBs for this enclave set-up will follow the Center structure of NASA. This discrimination makes sense, since each Center collects its own data and is usually only reporting to headquarters but not sharing with other Centers. All Centers are very strict about how data is published and used outside a Center environment. One privacy rule, for instance, requires that it is not possible to identify from which Center the data was derived. Other privacy rules will focus on the protection of the developers who submitted the data, the people who conducted the inspections, as well as details about the projects for which the inspection was conducted.

Possible research questions for case studies with PIBs, as well as privacy restrictions for the enclave which can be studied are listed in Table 2. A short description why the data is sensitive and which different nodes are in the enclave is included as well.

PIB research questions and nodes	Enclave privacy restrictions
What is a typical effort distribution over	People (i.e., names) involved in developing

development phases? How much effort does it take to correct a major / minor error? Nodes will be the different classes of a concrete NASA project. The enclave refinement for this case study is similar to levels 4+5 described in Table 4.	the class Owner of the class Center (i.e., name) developing the class Program or mission the class is/was developed for Used programming language → All restrictions focusing on the protection of specific people on the development team so they continue to share their data without having to fear that this will be held against them at any later point in time. Additional restrictions focusing on the protection of the integrity of a concrete Center or program
How are defects distributed over different project phases? What kind of defects are typically for each development phase? As nodes we will use the different NASA Centers. The enclave refinement for this case study is similar to levels 4+5 described in Table 4.	Program or mission the class is/was developed for Center (i.e., name) reporting Size of development team (i.e., # of people) Period of code development Used programming language → Restrictions focusing on the protection of the integrity of a concrete Center or program. Using team size and/or development year could allow to indirectly identifying the program/mission.
How does reuse of code influence the	Size of developed code (e.g., LoC)

defect density of a program? As nodes we will use the different (aerospace) projects. The enclave refinement for this case study is similar to levels 3-5 described in Table 4.	Size of project team (i.e., # of people) Program(s) (i.e., names) from which code was reused Current project phase (e.g., requirements, testing, finished/in use) Used programming language → All restrictions focusing on the protection of the integrity of a concrete program. Using the different information could indirectly allow 3rd parties to identify the original source of the data.

Table 2: Possible research questions vs. privacy restrictions for the software engineering domain in context of aerospace applications

Research, second year:

After we formalized and consolidated our knowledge on the impact of protocol rules on possible findings in an enclave, we will broaden our view. Therefore, we add new dimensions by applying our findings in:

A different domain, namely the medical domain in context of the WV hospitals, and

By refining our knowledge about the use of sub-enclaves. (i.e., using our recursive definition of enclaves)

Thereby, the focus of our studies is also placed on proofing the domain independent applicability of our enclave-based research approach. Data-sets for the case studies will be provided by WVU hospitals. In Table 3 we list possible research questions for the PIBs in the medical domain, as well as privacy restrictions for such an enclave. The table uses the same structure as for the software engineering domain above.

PIB research questions and nodes	Enclave privacy restrictions

How do treatments for a particular diagnosis vary across locations? What is the typical treatment for a particular diagnosis? Nodes will be community health clinics. The enclave refinement for this case study is similar to levels 2-5 described in Table 4.	? → Restrictions focusing on the protection of Healthcare providers, including both the specific clinic as well as the physicians and other healthcare workers. Restrictions on all patient identifiers in compliance with HIPAA including the general geographic location of the patient's home.
How does the presence of chronic conditions affect the choice of treatment? What are the characteristics of patients for whom standard treatment did not work? (chronic conditions such as diabetes, obesity, cardiovascular disease) Nodes will be patient sub categories based on chronic conditions. The enclave refinement for this case study is similar to levels 3-5 described in Table 4.	? → Restrictions focusing on the protection of patient data, physician identity, hospital unit providing the service and patient identifiers.
How is the treatment of a patient referred for specialized care affected by the source and timing of referrals? What impact does the stage of progression of the disease As nodes we would look at referring physician groups, and specialists to whom patients were referred. The enclave refinement	? → Restrictions on the identity of the referring the patient, all patient identifying information, including the geographic location of the referred patient.

for this case study is similar to levels 4-5

described in Table 4.

Table 3 Possible research questions vs. privacy restrictions for the medical domain in context of WVU hospitals

Research, third year:

Generalize our knowledge and develop open source enclave tool kit for different domains. Provide detailed application guidance for external users from academia, government, and industry.

Research, fourth year:

Repeat studies with third party user(s) in other domain(s)

Figure 1 summarizes the time-line for the proposed activities.

GANT CHART <<TO BE DONE>>
Figure 1: Project plan.

MISC NOTES. DO NOT DELTE!!!