# Model-based Discrimination Analysis: A Position Paper*

Qusai Ramadan
Amir Shayan Ahmadian
Daniel Strüber
University of Koblenz-Landau, Germany
{qramadan,ahmadian,strueber}@uni-koblenz.de

Jan Jürjens
University of Koblenz-Landau, Germany
Fraunhofer ISST, Dortmund, Germany
http://jan.jurjens.de

Steffen Staab
University of Koblenz-Landau, Germany
University of Southampton, UK
staab@uni-koblenz.de

## ABSTRACT

Decision-making software may exhibit biases due to hidden dependencies between protected characteristics and the data used as input for making decisions. To uncover such dependencies, we propose the development of a framework to support discrimination analysis during the system design phase, based on system models and available data.

## KEYWORDS

Software Fairness, UML, Model-based Analysis

## 1 INTRODUCTION

Advances in machine learning and the availability of vast amounts of data have enabled the rise of autonomous decision-making software, which is now used in various industries [15, 18]. Such software uses machine learning to reveal useful patterns and trends across large sets of data, in order to elucidate judgment rules and apply them to new cases. Many applications are socially sensitive: For instance, machine learning is now used to decide whether a sentenced person should be released, who is invited to a job interview, or which kind of medical treatment is offered to a patient.

There is a risk that such patterns, or indeed any automated decision making procedure, can be used to unlawfully discriminate against persons based on their protected characteristics, either intentionally or unintentionally. A protected characteristic is any personal information that should not be subject to discrimination in a decision-making process. For instance, the UK Equality Act 2010 and the German General Equal Treatment Act (AGG) 2006, define some legally protected characteristics, including age, gender, race, and sexual orientation. However, the protected characteristics are not limited to those listed by the laws and regulations. Depending on the considered business organization policies, other protected characteristics can be specified. For instance, citizenship is not considered as a protected characteristics in the Equality Act and the AGG, but it can act as one in the policies of a specific organization. For example, for loan decision-making, a bank may disallow discriminating between the customers based on their citizenship.

According to the recent literature [5, 11], available decision-making software is prone to illegal or unethical judgments that may lead to undesirable consequences such as reputation damage and law infringement. Often, to remain legally compliant, automated decision-making software avoids using protected characteristics as part of the input of the decision-making component. Unfortunately, these characteristics may still affect the analysis result: First, the actual input being used may contain data that resulted from processing protected characteristics, thus indirectly revealing signals about them. Second, while a given protected characteristic (e.g. gender) may not be not processed, other background data (e.g. educational background) can act as proxies for these characteristics.

A system developer without knowledge about these dependencies may easily develop a decision-making software which discriminates against protected characteristics. Worse, in some cases, such discrimination is inherently hard to detect due to hidden information flows in the system that indirectly leak a signal about protected characteristics to the output of a decision-making software. According to Barocas and Selbst [3, p. 1] *"[...] because the resulting discrimination is almost always an unintentional emergent property of the algorithm's use rather than a conscious choice by its programmers, it can be unusually hard to identify the source of the problem or to explain it to a court"*.

Existing works either focus on the machine learning algorithm [4, 14, 19] or on testing the overall system [9]. While the former can reduce known discrimination by processing the used data or model, it cannot uncover discrimination in the above mentioned sense, as it does not consider the algorithm's context. The latter treats the system as a black-box and studies the output behavior of a given system *after the fact*, that is, when it has been implemented.

In this paper, we argue that responsible behavior needs dedication and support from the early stages of the system design. System developers should be supported with tools to specify the protected characteristics, and to reason about hidden information flows between these characteristics and decisions. Detecting implicit flows of information is not a new topic, but a key challenge in security engineering [8, 10, 12, 17]. However, we are not aware of a model-based security analysis approach that supports the engineering of discrimination-aware information systems.

We address the following research question: *How can one uncover dependencies between protected characteristics and the output of a critical decision-making process during the system design phase?* We

*Produces the permission block, and copyright information


*FairWare'18, May 29, 2018, Gothenburg, Sweden*

ACM ISBN 978-1-4503-5746-3/18/05…$15.00
https://doi.org/10.1145/3194770.3194775

**Table 1: Personal Data**

| customer_ID | gender | income | high_Income | nationality | merchant | accounting |
|---|---|---|---|---|---|---|
| BA01 | 0 | 3000 | 0 | 0 | 1 | 1 |
| BA02 | 0 | 4500 | 1 | 1 | 1 | 1 |
| BA03 | 0 | 2500 | 0 | 1 | 0 | 1 |
| BA04 | 0 | 3200 | 1 | 1 | 0 | 1 |
| BA05 | 0 | 2900 | 0 | 1 | 0 | 1 |
| BA06 | 1 | 5000 | 1 | 0 | 1 | 1 |
| BA07 | 1 | 2450 | 0 | 0 | 1 | 0 |
| BA08 | 1 | 3600 | 1 | 1 | 1 | 1 |
| BA09 | 1 | 3100 | 1 | 0 | 1 | 1 |
| BA10 | 1 | 1800 | 0 | 1 | 0 | 1 |

**Table 2. Services Data**

| customer_ID | zero_Fee | vac_Ann |
|---|---|---|
| BA01 | 0 | 1 |
| BA02 | 1 | 0 |
| BA03 | 0 | 0 |
| BA04 | 0 | 0 |
| BA05 | 0 | 0 |
| BA06 | 0 | 1 |
| BA07 | 0 | 0 |
| BA08 | 1 | 0 |
| BA09 | 0 | 1 |
| BA10 | 1 | 0 |

**Table 3. Previous loan Requests**

| request_ID | customer_ID | amount | result |
|---|---|---|---|
| LR01 | BA01 | 10000 | 1 |
| LR02 | BA02 | 5000 | 1 |
| LR03 | BA03 | 3000 | 0 |
| LR04 | BA04 | 4000 | 1 |
| LR05 | BA05 | 12000 | 0 |
| LR06 | BA06 | 30000 | 1 |
| LR07 | BA07 | 18000 | 0 |
| LR08 | BA08 | 2000 | 1 |
| LR09 | BA09 | 6000 | 1 |
| LR10 | BA10 | 4000 | 0 |

**Figure 1: Bank database with customer data.**

propose the development of a model-based analysis framework. The key idea is to analyze the system models and available data to uncover leaks of protected characteristics. In particular, we aim to address proxy discrimination [7], a frequent discrimination phenomenon: for example, a gender background may not used as input to the decision-making software, but if the males are more likely to have a higher income than females, then the income may act as a proxy for the gender. Consequently, approving that a protected characteristic is not processed at all, either directly or indirectly, is not sufficient to minimize discrimination. To address this issue, we suggest performing a statistical analysis using the result of the information flow analysis and a database of available data from the system's context. Based on the results of the analysis, we can raise awareness of possible discrimination in the system.

Avoiding the use of proxy characteristics is generally not always possible or even desirable. For example, the developers may want to use proxy characteristics together with the protected characteristics in order to actively re-balance the output such that no group is discriminated against. For these reasons, the analysis results will be used to generate warnings that might require further analysis to see whether the system discriminates for real.

The paper is organized as follows. Sect. 2 provides a motivating example. Sect. 3 gives a problem statement. Sect. 4 introduces a roadmap to our proposed framework.

## 2 MOTIVATING EXAMPLE

The following example is inspired by real critical decision-making processes in banking systems.

Consider a bank interested in leveraging automatic decision-making to provide certain services to different customers. Three example services are: (i) Apply for a Loan, which is available for all customers who want to apply for a loan. (ii) zero-Fee Money Exchange, which allows customers to exchange money without paying extra fees. This service is available for the customers working as international merchants. (iii) Announcements about Job Vacancies, which sends announcements on job vacancies to its customers. This service is available only for national customers with an educational background in accounting. The bank has different discrimination policies when offering services to specific customers. For example, for loan decision-making, the bank policies disallow discriminating between the customers based on their citizenship, while for other services it is considered acceptable.

Among these three services, Apply for a loan service is particularly critical. When developing a system for automating it, the decision whether a loan is risky or safe is to be generated based on historical observations. In our case, these observations come from an existing database that stores data about the bank's customers. Fig. 1 shows the database considered in this paper. The database contains information about 10 customers with equal numbers of females and males.

The Personal Data table in Fig. 1 stores personal data about the customers. The gender is a boolean attribute where 1 refers to males and 0 to females. The income and high_Income represent financial information about the customers. The high_Income is a boolean attribute representing whether a customer receives income above a certain threshold. If a customer earns more than 3K Euro per month, the corresponding cell shows a 1, else a 0. The nationality column shows information about the customer's citizenship. If a costumer is an international customer, the corresponding cell shows a 1, otherwise a 0. The merchant and accounting attributes shows the job and the educational background information of the costumers, respectively. For simplicity, we considered one job type (merchant) and one educational background category (accounting). If a customer works as a merchant or has a an educational background in accounting, the corresponding cells include a 1, else a 0.

Customers may apply for one or multiple loan requests at one time. In Fig. 1, the Previous loan_request table stores data about previous requests, including the request_ID, amount and result (i.e., safe or risky). The result is a boolean attribute representing whether the request was considered as safe (1) or risky (0). A service in the bank can be available to many customers, while a customer may receive many services. The Service data table stores data about the banks' services and their availability to a customer. The zero_Fee and vac_Ann attributes refer to the zero_Fee Money Exchange service and the Announcement about Job Vacancies service, respectively. If the zero_Fee Money Exchange service is available to a customer, the corresponding cell shows a 1, else a 0. The Announcement about Job Vacancies service is handled similarly.

**The challenge.** As system analysts, we need to comply with the bank's discrimination policies. We focus on the following question: Does the decision-making software possibly discriminate between the customers based on their *citizenship* (i.e., national vs international)?

**System models.** As a first step towards solving this challenge, we need to know which of the customer data in Fig. 1 is used as input

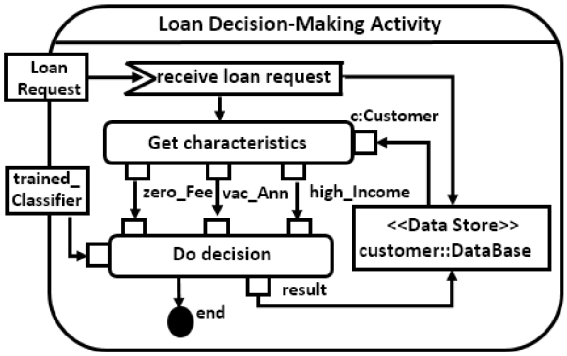


**Figure 2: Activity diagram describing the loan decision-making process.**

to the decision-making software. Unfortunately, the database is a static representation for the data structure and it does not tell which data is being processed by the decision-making software. Generally, dealing with the system as a black box will not help us to identify the input of a decision making software. Consequently, we will not be able to address our challenge.

In a complex decision-making system, identifying the data used to inform the decisions is an effort- and time-consuming task when done manually on the basis of an existing implementation. In critical domains such as finance, the system is typically modeled before it is implemented, providing a high-level specification in which system components with their inputs and outputs can be analyzed. Fig. 2 shows a simple UML activity diagram [16] for the loan decision-making process. The activity starts by receiving a loan request event. The received loan request is automatically stored in the customer database. Afterwards, the get characteristics action is invoked. This action consumes three data objects as input, namely, zero_Fee, vac_Ann and high_Income. The first two data objects respectively represent the availability of the zero-fee Money Exchange and Announcements about Job Vacancies services to the loan applicant. The high_Income object specifies whether the applicant has a high income or not. Similarly, these data objects are used as input for the Do decision action.

Rectangles on the boundaries of an activity diagram represent data objects that are either input data from or output data provided to another activity in the system. For example, the trained_classifier data object is the result of training a machine learning classifier which is actually an output from another activity in the system. The output of the Do decision action is a data object called result which shows whether a loan request is *safe* or *risky*. Since the result data object represents a critical decision we will call it a *critical data object* in what follows.

**Statistical analysis.** Based on the activity diagram shown in Fig. 2, the decision-making software processes the zero_Fee, vac_Ann and high_Income data objects directly to produce the critical data object. Although the *citizenship* data was not used as part of the direct input, we knew that some of these data may act as proxies to the citizenship data. Given the database in Fig. 1, we can perform a statistical analysis to uncover dependencies between the direct input of our decision-making software and the customers' citizenship data. In Table 1, the column P(international) shows the conditional probability of the customer being international, given the data objects being directly processed by the decision-making software. For instance, the first row shows the probability of a customer being an international given that (i) he does not have a high income, and (ii) the zero_Fee Money Exchange and the Announcement about Job Vacancies services are not available to him, as 66.67%.

**Table 1: Conditional probabilities of directly processed data.**

| P(international) | zero_Fee | vac_Ann | high_Income |
|---|---|---|---|
| 66.67% | 0 | 0 | 0 |
| 100.00% | 1 | 0 | 0 |
| 0.00% | 1 | 1 | 0 |
| 100.00% | 1 | 0 | 1 |
| 0.00% | 0 | 1 | 0 |
| 0.00% | 0 | 1 | 1 |
| 100.00% | 0 | 0 | 1 |

Assuming a normally distributed dataset, we generally expect mid-range values for the probability of a customer being an international. Probabilities that equal 100% or 0% would be more of a surprise. However, in Table 1, this case happens when the zero_Fee Money Exchange service or the Announcement about Job Vacancies service is available. For example, the second row in the table tells us that their is a maximal correlation between being an international customer and receiving a zero_Fee Money Exchange service.

Although it is clear that both the zero_Fee and vac_Ann data objects can act as proxies for the customers' *citizenship* data, still we have no explicit explanation for their probabilities. For example, it's not clear why the relation between the zero_Fee service and being an international customer is so high.

**Information-flow analysis.** In the following we discuss two possible explanations: First, the zero_Fee and the vac_Ann data objects could be strongly correlated with the citizenship data due to a societal fact (e.g., knowing that a person is working as a taxi driver in Saudi Arabia, one can directly say that he is a male). Such situation is not related to anomalous distributions in the database and one can not even avoid them. Second, zero_Fee and the vac_Ann could be derived data resulting from processing the citizenship information. As a result, the derived data still retain strong signals about the citizenship information.

Generally, in any software system that processes users data to provide them with services, the result of processing the users' data are new data. We call these data, which is generated from processing other data, *derived data*. Information about whether a data object is derived or not can be represented in the database schema. The database schema describes the structure of database tables and provides meta-data about the database data such as the data value type (e.g., Integer and String) and whether a data is derived or not. The literature provides many modeling languages for representing the database schema such as the entity relationship diagram [6] and the UML class diagram [16]. The UML class diagram is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among objects [16]. However, one

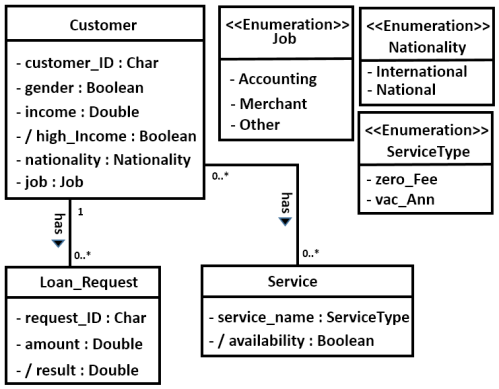


**Figure 3: Database schema, specified using a class diagram.**

can also use the UML class diagram to describe the structure of a database tables, their relationships, their columns as attributes and metadata about each attribute such as the input type (e.g., integer or string) and derived or not derived..

Fig. 3 shows a class diagram that describes the underlying schema of the database in Fig. 1. Derived data are visually represented using the "/" character. For example, the high_Income , in Fig. 3, is a derived data object. Using this information, we can address the remaining question: *why are zero_Fee and vac_Ann strongly correlated with the citizenship information?*

Although the database schema shows whether a data object is derived, it does not tell how it is derived. This information can be found in the activity diagrams. Therefore, as a business analyst, we need to find out and analyze the activity diagrams that describes how a certain data object is derived. Such type of analysis is called information-flow analysis. The information-flow analysis allows to: First, understand how the data are propagated in the system. Second, uncover critical leakage for *sensitive* data [13]. With an analysis for the information flows of the targeted decision-making software, the analyst can minimize the risk of having indirect discrimination against protected characteristic.

Fig. 4 is an activity diagram describing how the value of the zero_Fee data object is derived. The activity starts by retrieving all data instances of type Customer. The retrieved instances will be listed in a buffer. Then a loop of processes that works over all the listed instances will start. The Get Job and Citizenship info action receives the customer_ID as input and returns the corresponding job and citizenship data to the customer_ID. The result of of this action will be aggregated in a one data object called result. The result will be verified by the exclusive gateway. If the customer is working as an international merchant, the Update the zero_Fee Exchange action will be invoked. This action will update the value of the zero_Fee data object. Otherwise, the process for that instance will be terminated. The loop will be terminated when there are no more instances in the list.

Based on the description of the above activity, we infer that the zero_Fee attribute is derived from processing the job and citizenship data. More precisely, the value of the zero_Fee depends on whether a customer is international and working as a merchant.

For brevity, we do not include the activity diagrams showing how the vac_Ann and high_Income data objects are derived in this paper. Instead, we will refer to the early description of our motivating example. First, the Announcement about Job Vacancies, as described in the motivating example, is an available service only for any national customer who has an educational background in accounting. The vac_Ann attribute that shows the availability of this service is specified as derived, because its value depends on whether a customer is a national customer and has an educational background in accounting. Second, the high_Inc data object, as described earlier in this paper, is a derived data object because it value depends on whether the income of a customer exceeds a certain average.

Fig. 5 represents the flow of data in the loan decision-making software. The flow is represented as a tree of connected nodes, where each node represents a data object. The figure shows the directly and the indirectly processed data objects and the relations between them.

Although the customers' *citizenship* data was not used directly as input to the decision-making software, Fig. 5 shows that there is indirect leakage for the citizenship data to the result data object. This is because both the zero_Fee and the vac_Ann are resulting from processing the citizenship data. As a result, we now have an explicit explanation to the probabilities in Table 1. More precisely, for example, the reason for having a high correlation between the zero_Fee and the international is coming from the fact that the zero_Fee represents a derived data object whose value is derived from processing the citizenship and the working data of a customer.

Since the information flow shows that there is indirect leakage for the citizenship data to the input of the decision-making software, we as an analyst can conclude that this process is illegal with respect to the bank's policies.

**Discussion** In the above case, the data flows analysis helped us to uncover indirect (i.e., hidden) processing for protected characteristic. To this end, consider, for example, that our intention in the previous section was to uncover dependencies between the output of the decision-making software and the *gender* instead of the *citizenship*. Fig. 5 shows evidence that there is no processing either directly or indirectly for the gender. Our question now: *Is this sufficient to believe that the output of a decision-making software has no dependencies with the gender?*

According to the literature the input to a decision-making software may include data that acts as proxies for a protected characteristic. Discrimination arising due to use of data correlated to protected characteristic is referred to as *discrimination by proxy* [7]. For example, if a database shows that females are more likely to have an educational background in *accounting* than the males. Then *accounting* can act as proxy for the *gender*. Hence, an evidence that a protected characteristic was not processed at all (i.e., either directly or indirectly) does not necessary mean that their is no dependency between the output of a decision-making software and that protected characteristic.

Since today's systems store much data about their customers, a single protected characteristic may have many proxies. Finding all the possible proxies and reasoning about them is a difficult task. To minimize the risk of discrimination against protected characteristic,

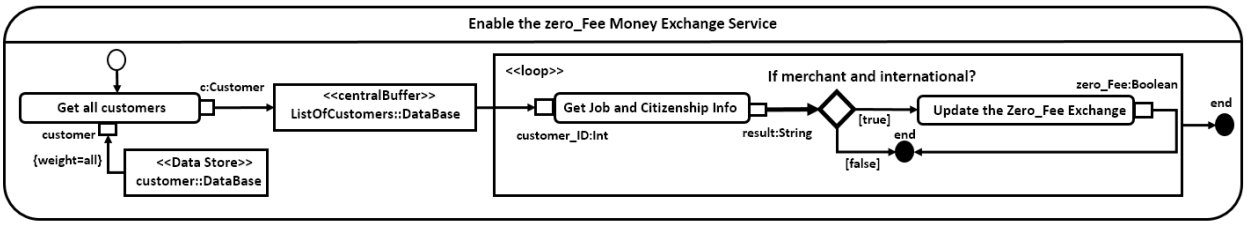


**Figure 4: Activity diagram describing how the value of the *zero_Fee* data object is derived.**

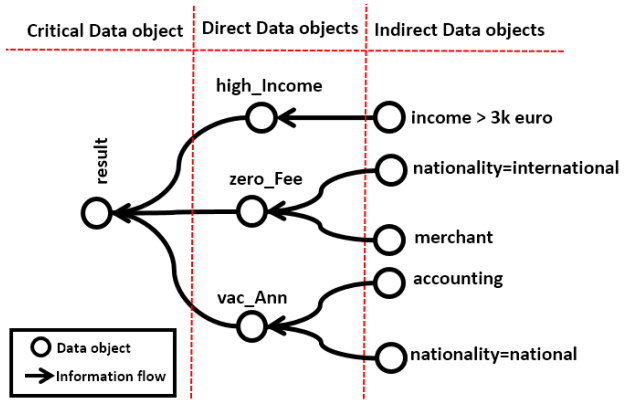


**Figure 5: Data flows in the example.**

we can reuse the data flows analysis results to study the correlation between the leaves nodes in the resulting data-flow tree in Fig. 5 and the protected characteristic (i.e., gender).

Based on the database in Fig. 1, we calculated the conditional probabilities of being female given the leave nodes in the data-flow tree from Fig. 5. The conditional probabilities are summarized in Table 2. For example, row number 2 in the table represents the probability P(Female | ¬(">3k" €∩ national ∩ merchant) ∩ accounting). In other words, row number 2 shows the probability of being female given the that the customer: (i) does not receive a high_income. (ii) is not a national customer. (iii) does not work as a merchant. (iv) has an educational background in accounting.

By looking to the database in Fig. 1, we can find that 3 out of 10 entries (i.e., rows) in the database match the condition in row number 2. These entries are highlighted in gray color in Table 1 from Fig. 1. Two out of these three cases belongs to females. Therefore, as shown in Table 2, the probability of being a female given the specified condition in row number 2 is 66.67%. Since the majority in this context (i.e., given a national customer with educational background in accounting and the other data objects are not true) are females, we can conclude that the nationality and the education background in this context can act as a proxy for the *gender*.

In conclusion, the goal of the information-flow analysis is twofold: First, it allowed us to uncover indirect leakage for protected data. Second, it helped us to uncover situations where the used data can act as a proxy for a protected characteristic.

## 3 PROBLEM STATEMENT

To remain legally compliant with the laws and regulations and to avoid expensive fixes, the developers of a system must minimize the source of possible discrimination already during the design of an automated decision-making software. Relying only on a given database will not help the analyst to uncover dependencies between the output and a given protected characteristic. A main source for this problem is the existence of hidden critical information flows that indirectly leak signals about protected characteristic to the input of a decision-making software. On the other hand, testing does not help to uncover problematic dependencies at the design time, since it assumes a complete implementation of the system, the development of which is an effort- and time-consuming task.

A solution for this issue is to analyze the system design model. However, analyzing the system design model manually is a difficult and error-prone task because information about how the data are propagated in the system and how they are related with each other are hidden and distributed in multiple types of diagrams. In addition, the system model alone is not sufficient for detecting statistical dependencies between used data and protected characteristics. The detection of these dependencies requires to apply statistical analysis to the available information as well.

**Roles:** The following roles are assumed: First, *domain experts* who clarify which kinds of discrimination are allowed (e.g. age discrimination for life insurance) and which ones are not (e.g. age discrimination for hiring decisions). Second, *software analyst/developer* who has a good background in modeling and some expertise in statistics.

**Input:** Three types of inputs are assumed: First, a *requirements document* containing discrimination-aware requirements. During the requirements elicitation, the domain expert, based on the organizational policies and laws, can identify *what can should be protected* and in *which context*.

**Table 2: Examples of the conditional probabilities of the gender with the indirect processed data**

| P(Female) | ">3k" € | nationality | merchant | accounting |
|---|---|---|---|---|
| 50.00% | 1 | 1 | 1 | 1 |
| 66.67% | 0 | 1 | 0 | 1 |
| 0.00% | 0 | 0 | 1 | 0 |
| 100.00% | 0 | 0 | 1 | 1 |
| 100.00% | 1 | 1 | 0 | 1 |
| 0.00% | 1 | 0 | 1 | 1 |

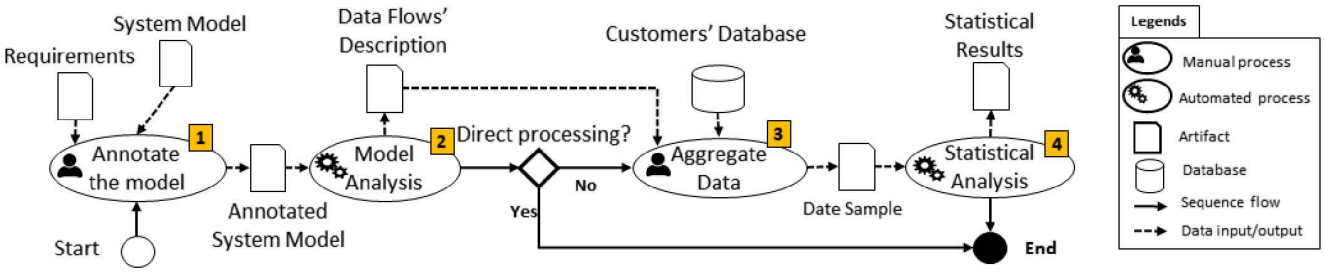


**Figure 6: High-level overview of a model-based discrimination analysis framework.**

Second, a *system model* represented by the Unified Modeling Language (UML) [16]. UML permits the developers to model different aspects about the system. The structural aspects of the system are specified by using class diagrams, while the behavior aspects are specified by using, for example, activity diagrams. While the use of system models in practice varies between different software domains, they can be a key enabler for important tasks of high business value, such as the discrimination analysis presented here.

Third, a *database* of historical data. The database needs to contain data for protected as well as other characteristics, in order to support the detection of dependencies between them. To support reasoning about the system design based on the database, we also assume a mapping from the system model to the database schema, which could be given by the user or determined (semi-)automatically.

**Output:** The output can be delivered on different levels of granularity. Most fine-grained would be a report of all statistic analysis outputs, whose interpretation is left to the analyst. To reduce the user involvement, more coarse-grained options require pre-defined thresholds to automatically distinguish between critical and uncritical statistical effects. Problematic information flows and elements can then be highlighted inside the model.

## 4 ROADMAP: DEVELOPING A FRAMEWORK FOR MODEL-BASED DISCRIMINATION ANALYSIS

To address this problem, we propose to develop a model-based analysis framework, an overview being shown in Fig 6. We outline the roadmap to the realization of our framework.

**Step 1: Supporting model annotations.** To allow for an automated analysis for the system models, the analyst must be able to enrich the system models with discrimination-related information such as what are the protected characteristics and what are the critical data objects (i.e., the data object that should not has dependencies with protected data) in our system. For this, we plan to extend the privacy UML profile in [2] to allow annotating the system model with discrimination-related information. The output of this phase will be an annotated system design model.

**Step 2: Providing an automated analysis.** This phase takes as input an annotated UML model from the previous phase to uncover about hidden and critical information flows in that system model.

In this phase, all the data objects that will be directly or indirectly processed for generating a critical-annotated data object will be represented as a tree of connected nodes, where each node in the tree represent a data object. The following is a a detailed description about how the tree will be generated: First, the critical-annotated data object in the model will be represented as a root node. Second, the input to the action that is directly responsible about generating the critical data object will be listed as children nodes for the root node in the tree. The children nodes of the root node represent the data objects that are used directly for generating the critical data object. Third, each child node will be verified. If the node represents a derived data object, then all the data objects that have been directly consumed to generate that derived data object will be listed as children nodes for its corresponding node in the tree. The third step will be repeated until either the protected characteristic appears (e.g., gender) as a node in the tree or until there is no more derived data objects in the resulted tree.

Since information about whether a data object is derived and how it is derived are generally hidden and distributed in multiple UML diagrams, analyzing the information flow is difficult. For example, Fig. 2, Fig. 3 and Fig. 4 are not internally connected. Therefore, to automate the process of information flow analysis, we plan to: First, provide a precise semantics for the proposed annotations. Second, extend the formalisms in [13] by formalizing the information flow inside the UML activity diagram and formalizing the information flow analysis in a way that allows processing different UML diagrams in an integrated manner. The work in [13] provides formalizations to different types of UML diagrams, including activity diagrams. However, the activity diagram formalism in [13] did not consider data flow, which was only added to activity diagrams in UML 2.0 [16].

To implement the process of information flow analysis we are planning to extend a tool analysis support called CARiSMA [1]. The output of this phase will be information about how the data are propagated in the system.

**Step 3: Supporting data aggregation.** Due to the proxy discrimination challenge, an approval that protected characteristics are not processed at all (i.e., directly or indirectly) is not sufficient to minimize the risk of discrimination: a statistical analysis is needed. Instead of considering all possible subsets of characteristics in the database, our proposed framework suggests to focus on correlations between the protected characteristic and the data that are processed to generate the critical decision in the system model.

The analyst will generate a data sample from the database. Each record in the sample will represent an observation from historical data and contains information about the protected characteristics, the directly and indirectly processed data from Phase 2.

**Step 4: Performing statistical analysis.** We want to support the analyst in using statistics to reason about the correlation between the protected characteristic and other data in the generated sample. The statistics will help the analyst to uncover data that can act as proxies for a protected characteristic.

## ACKNOWLEDGMENT

This research is partially supported by the Deutsche Akademische Austauschdienst (DAAD), the project "Engineering Responsible Information Systems" financed by the University of Koblenz-Landau and a partial funding is acknowledged from the project SecVolution@Run-time (JU 2734/2-2 and SCHN 1072/4-2) which is part of the priority programme SPP 1593 "Design For Future - Managed Software Evolution" of the German Science Foundation (DFG).

## REFERENCES

[1] Amir Shayan Ahmadian, Sven Peldszus, Qusai Ramadan, and Jan Jürjens. 2017. Model-based privacy and security analysis with CARiSMA. In *Proceedings of the 2017 11th Joint Meeting on Foundations of Software Engineering*. ACM, 989–993.

[2] Amir Shayan Ahmadian, Daniel Strüber, Volker Riediger, and Jan Jürjens. 2017. Model-based privacy analysis in industrial ecosystems. In *European Conference on Modelling Foundations and Applications*. Springer, 215–231.

[3] Solon Barocas and Andrew D Selbst. 2016. Big data's disparate impact. (2016).

[4] Toon Calders and Sicco Verwer. 2010. Three naive Bayes approaches for discrimination-free classification. *Data Mining and Knowledge Discovery* 21, 2 (2010), 277–292.

[5] Laura Carmichael, Sophie Stalla-Bourdillon, and Steffen Staab. 2016. Data mining and automated discrimination: a mixed legal/technical perspective. *IEEE Intelligent Systems* 31, 6 (2016), 51–55.

[6] Peter Pin-Shan Chen. 1988. The entity-relationship modelâĂŤtoward a unified view of data. In *Readings in artificial intelligence and databases*. Elsevier, 98–111.

[7] Anupam Datta, Matt Fredrikson, Gihyuk Ko, Piotr Mardziel, and Shayak Sen. 2017. Proxy Non-Discrimination in Data-Driven Systems. *arXiv preprint arXiv:1707.08120* (2017).

[8] Dorothy E Denning. 1976. A lattice model of secure information flow. *Communications of tshe ACM* 19, 5 (1976), 236–243.

[9] Sainyam Galhotra, Yuriy Brun, and Alexandra Meliou. 2017. Fairness testing: testing software for discrimination. In *Proceedings of the 2017 11th Joint Meeting on Foundations of Software Engineering*. ACM, 498–510.

[10] Joseph A Goguen and José Meseguer. 1984. Unwinding and inference control. In *Security and Privacy, 1984 IEEE Symposium on*. IEEE, 75–75.

[11] Sara Hajian, Josep Domingo-Ferrer, Anna Monreale, Dino Pedreschi, and Fosca Giannotti. 2015. Discrimination-and privacy-aware patterns. *Data Mining and Knowledge Discovery* 29, 6 (2015), 1733–1782.

[12] Jan Jürjens. 2000. Secure Information Flow for Concurrent Processes, See [17], 395–409. DOI: http://dx.doi.org/10.1007/3-540-44618-4_29

[13] Jan Jürjens. 2005. *Secure systems development with UML*. Springer Science & Business Media.

[14] Faisal Kamiran, Toon Calders, and Mykola Pechenizkiy. 2010. Discrimination aware decision tree learning. In *Data Mining (ICDM), 2010 IEEE 10th International Conference on*. IEEE, 869–874.

[15] Toshihiro Kamishima, Shotaro Akaho, and Jun Sakuma. 2011. Fairness-aware learning through regularization approach. In *Data Mining Workshops (ICDMW), 2011 IEEE 11th International Conference on*. IEEE, 643–650.

[16] Object Management Group (OMG). 2011. UML 2.5 Superstructure Specification. (2011).

[17] Catuscia Palamidessi (Ed.). 2000. *CONCUR 2000 - Concurrency Theory, 11th International Conference, University Park, PA, USA, August 22-25, 2000, Proceedings*. Lecture Notes in Computer Science, Vol. 1877. Springer. DOI: http://dx.doi.org/10.1007/3-540-44618-4

[18] Frank Pasquale. 2015. *The black box society: The secret algorithms that control money and information*. Harvard University Press.

[19] Muhammad Bilal Zafar, Isabel Valera, Manuel Gomez Rodriguez, and Krishna P Gummadi. 2015. Learning fair classifiers. *arXiv preprint arXiv:1507.05259* (2015).