PRIVACY POLICY INFERENCE OF USER-UPLOADED IMAGES ON CONTENT SHARING SITES

[1]Ch. Ramesh and **[2]B. Sowjanya.**
[1]M.E. Student, [2]Asst. Professor
[1, 2] Department of Computer Science and Engineering, Methodist College of Engineering and Technology, Hyderabad, India,
Email: [1]chin.ramesh01@gmail.com, [2]Sowjanyayk6@gmail.com

ABSTRACT

With the increasing volume of images users share through social sites, maintaining privacy has become a major problem, as demonstrated by a recent wave of publicized incidents where users inadvertently shared personal information. In light of these incidents, the need of tools to help users control access to their shared content is apparent. Toward addressing this need, we propose an Adaptive Privacy Policy Prediction (A3P) system to help users compose privacy settings for their images. We examine the role of social context, image content, and metadata as possible indicators of users' privacy preferences. We propose a two-level framework which according to the user's available history on the site determines the best available privacy policy for the user's images being uploaded. Our solution relies on an image classification framework for image categories which may be associated with similar policies, and on a policy prediction algorithm to automatically generate a policy for each newly uploaded image, also according to users' social features. Over time, the generated policies will follow the evolution of users' privacy attitude. We provide the results of our extensive evaluation over 5,000 policies, which demonstrate the effectiveness of our system, with prediction accuracies over 90 percent.

Index Terms— A3P, social context, image content, metadata

I. INTRODUCTION

IMAGES are now one of the key enablers of users' connectivity. Sharing takes place both among previously established groups of known people or social circles (e g., Google+, Flickr or Picasa), and also increasingly with people outside the users social circles, for purposes of social discovery-to help them identify new peers and learn about peers interests and social surroundings. However, semantically rich images may reveal content sensitive information. Consider a photo of a student's 2012 graduation ceremony, for example. It could be shared within a Google+ circle or Flickr group, but may unnecessarily expose the students BA pos family members and other friends. Sharing images within online content sharing sites, therefore, may quickly lead to unwanted disclosure and privacy violations [3], [24]. Further, the persistent nature of online media makes it possible for other users to collect rich aggregated information about the owner of the published content and the subjects in the published content [3], [20], [24]. The aggregated information can result in unexpected exposure of one's social environment and lead to abuse of one's personal information.

ORGANIZATION PROFILE:

Software Solutions is an IT solution provider for a dynamic environment where business and technology strategies converge. Their approach focuses on new ways of business combining IT innovation and adoption while also leveraging an organization's current IT assets. Their work with large global corporations and new products or services and to implement prudent business and technology strategies in today's environment. Although buyers leave positive feedback ratings, they express some disappointment and negativeness in free text feedback comments [3], often towards specific aspects of transactions.

Xxxxxxx's RANGE OF EXPERTISE INCLUDES:

- Software Development Services
- Engineering Services
- Systems Integration
- Customer Relationship Management
- Product Development
- Electronic Commerce
- Consulting
- IT Outsourcing

We apply technology with innovation and responsibility to achieve two broad objectives:

- Effectively address the business issues our customers face today.
- Generate new opportunities that will help them stay ahead in the future.

THIS APPROACH RESTS ON:

- A strategy where we architect, integrate and manage technology services and solutions - we call it AIM for success.
- A robust offshore development methodology and reduced demand on customer resources.
- A focus on the use of reusable frameworks to provide cost and times benefits.

They combine the best people, processes and technology to achieve excellent results - consistency. We offer customers the advantages of:

SPEED:

They understand the importance of timing, of getting there before the competition. A rich portfolio of reusable, modular frameworks helps jump-start projects. Tried and tested methodology ensures that we follow a predictable, low - risk path to achieve results. Our track record is testimony to complex projects delivered within and evens before schedule.

EXPERTISE:

Our teams combine cutting edge technology skills with rich domain expertise. What's equally important - they share a strong customer orientation that means they actually start by listening to the customer. They're focused on coming up with solutions that serve customer requirements today and anticipate future needs.

A FULL SERVICE PORTFOLIO:

They offer customers the advantage of being able to Architect, integrate and manage technology services. This means that they can rely on one, fully accountable source instead of trying to integrate disparate multi-vendor solutions.

SERVICES:

Xxx is providing its services to companies which are in the field of production, quality control etc with their rich expertise and experience and information technology they are in best position to provide software solutions to distinct business requirements.

II. PURPOSE OF THE PROJECT

- Sharing images within online content sharing sites, therefore, may quickly lead to unwanted disclosure and privacy violations.
- Further, the persistent nature of online media makes it possible for other users to collect rich aggregated information about the owner of the published content and the subjects in the published content.
- The aggregated information can result in unexpected exposure of one's social environment and lead to abuse of one's personal information.

III. PROBLEM IN EXISTING SYSTEM

1. The costs and complexities involved generally increase with the number of the decryption keys to be shared.
2. The encryption key and decryption key are different in public key encryption.

SOLUTION OF THESE PROBLEMS

The above system has some drawbacks

- Most content sharing websites allow users to enter their privacy preferences. Unfortunately, recent studies have shown that users struggle to set up and maintain such privacy settings.
- One of the main reasons provided is that given the amount of shared information this process can be tedious and error-prone. Therefore, many have acknowledged the need of policy recommendation systems which can assist users to easily and properly configure privacy settings.

IV. SYSTEM ANALYSIS

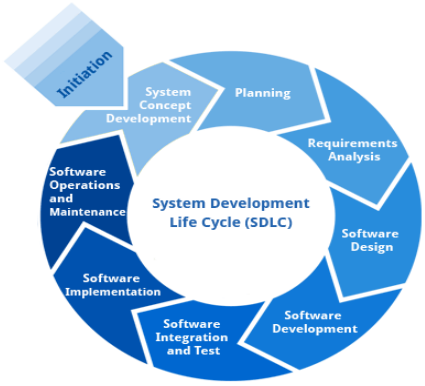
INTRODUCTION

Software Development Life Cycle:-

There is various software development approaches defined and designed which are used/employed during development process of software, these approaches are also referred as "Software Development Process Models". Each process model follows a particular life cycle in order to ensure success in process of software development.

Requirements

Business requirements are gathered in this phase. This phase is the main focus of the project managers and stake holders. Meetings with managers, stake holders and users are held in order to determine the requirements. Who is going to use the system? How will they use the system?



What data should be input into the system? What data should be output by the system? These are general questions that get answered during a requirements gathering phase. This produces a nice big list of functionality that the system should provide, which describes functions the system should perform, business logic that processes data, what data is stored and used by the system, and how the user interface should work. The overall result is the system as a whole and how it performs, not how it is actually going to do it.

Design

The software system design is produced from the results of the requirements phase. Architects have the ball in their court during this phase and this is the phase in which their focus lies. This is where the details on how the system will work is produced. Architecture, including hardware and software, communication, software design (UML is produced here) are all part of the deliverables of a design phase.

Implementation

Code is produced from the deliverables of the design phase during implementation, and this is the longest phase of the software development life cycle. For a developer, this is the main focus of the life cycle because this is where the code is produced. Implementation my overlap with both the design and testing phases. Many tools exists (CASE tools) to actually automate the production of code using information gathered and produced during the design phase.

Testing

During testing, the implementation is tested against the requirements to make sure that the product is actually solving the needs addressed and gathered during the requirements phase. Unit tests and system/acceptance tests are done during this phase. Unit tests act on a specific component of the system, while system tests act on the system as a whole.

So in a nutshell, that is a very basic overview of the general software development life cycle model. Now let's delve into some of the traditional and widely used variations.

SDLC METHDOLOGIES

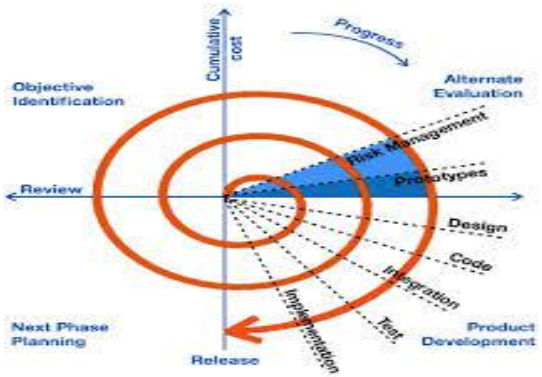
This document play a vital role in the development of life cycle (SDLC) as it describes the complete requirement of the system.

It means for use by developers and will be the basic during testing phase. Any changes made to the requirements in the future will have to go through formal change approval process.

SPIRAL MODEL was defined by Barry Boehm in his 1988 article, "A spiral Model of Software Development and Enhancement. This model was not the first model to discuss iterative development, but it was the first model to explain why the iteration models.

As originally envisioned, the iterations were typically 6 months to 2 years long. Each phase starts with a design goal and ends with a client reviewing the progress thus far. Analysis and engineering efforts are applied at each phase of the project, with an eye toward the end goal of the project.

The following diagram shows how a spiral model acts like:



The steps for Spiral Model can be generalized as follows:

- The new system requirements are defined in as much details as possible. This usually involves interviewing a number of users representing all the external or internal users and other aspects of the existing system.
- A preliminary design is created for the new system.
- A first prototype of the new system is constructed from the preliminary design. This is usually a scaled-down system, and represents an approximation of the characteristics of the final product.
- A second prototype is evolved by a fourfold procedure:
    1. Evaluating the first prototype in terms of its strengths, weakness, and risks.
    2. Defining the requirements of the second prototype.
    3. Planning a designing the second prototype.
    4. Constructing and testing the second prototype.
- At the customer option, the entire project can be aborted if the risk is deemed too great. Risk factors might involve development cost overruns, operating-cost miscalculation, or any other factor that could, in the customer's judgment, result in a less-than-satisfactory final product.
- The existing prototype is evaluated in the same manner as was the previous prototype, and if necessary, another prototype is developed from it according to the fourfold procedure outlined above.
- The preceding steps are iterated until the customer is satisfied that the refined prototype represents the final product desired.
- The final system is constructed, based on the refined prototype.
- The final system is thoroughly evaluated and tested. Routine maintenance is carried on a continuing basis to prevent large scale failures and to minimize down time.

STUDY OF THE SYSTEM

In the flexibility of uses the interface has been developed a graphics concepts in mind, associated through a browser interface. The GUI's at the top level has been categorized as follows

1. Administrative User Interface Design
2. The Operational and Generic User Interface Design

The administrative user interface concentrates on the consistent information that is practically, part of the organizational activities and which needs proper authentication for the data collection. The Interface helps the administration with all the transactional states like data insertion, data deletion, and data updating along with executive data search capabilities.

The operational and generic user interface helps the users upon the system in transactions through the existing data and required services. The operational user interface also helps the ordinary users in managing their own information helps the ordinary users in managing their own information in a customized manner as per the assisted flexibilities.

Modules Involved

- System Construction Module
- Content-Based Classification
- Metadata-Based Classification
- Adaptive Policy Prediction

MODULES DESCSRIPTION:

System Construction Module

The A3P system consists of two main components: A3P-coreand A3P-social. The overall data flow is the following. When a user uploads an image, the image will be first sent to the A3P-core. The A3P-core classifies the image and determines whether there is a need to invoke the A3P-social.In most cases, the A3P-core predicts policies for the users directly based on their historical behavior. If one of the

following two cases is verified true, A3P-core will invoke A3P social:(i) The user does not have enough data for the type of the uploaded image to conduct policy prediction; (ii) TheA3P-core detects the recent major changes among the user's community about their privacy practices along with user's increase of social networking activities (addition of new friends, new posts on one's profile etc).

Content-Based Classification

To obtain groups of images that may be associated with similar privacy preferences, we propose a hierarchical image classification which classifies images first based on their contents and then refine each category into subcategories based on their metadata. Images that do not have metadata will be grouped only by content. Such a hierarchical classification gives a higher priority to image content and minimizes the influence of missing tags. Note that it is possible that some images are included in multiple categories as long as they contain the typical content features or metadata of those categories.

Our approach to content-based classification is based on an efficient and yet accurate image similarity approach. Specifically, our classification algorithm compares image signatures defined based on quantified and sanitized version of Haar wavelet transformation. For each image, the wavelet transform encodes frequency and spatial information related to image color, size, invariant transform, shape, texture, symmetry, etc. Then, a small number of coefficients are selected to form the signature of the image. The content similarity among images is then determined by the distance among their image signatures.

Metadata-Based Classification

The metadata-based classification groups images into subcategories under aforementioned baseline categories. The process consists of three main steps. The first step is to extract keywords from the metadata associated with an image.

The metadata considered in outwork are tags, captions, and comments. The second step is to derive a representative hyponym (denoted as h) from each metadata vector. The third step is to find a subcategory that an image belongs to. This is an incremental procedure. At the beginning, the first image forms a subcategory as itself and the representative hyponyms of the image becomes the subcategory's representative hyponyms.

Adaptive Policy Prediction

The policy prediction algorithm provides a predicted policy of a newly uploaded image to the user for his/her reference. More importantly, the predicted policy will reflect the possible changes of a user's privacy concerns. The prediction process consists of three main phases: (i) policy normalization ;(ii) policy mining; and (iii) policy prediction.

b) PROPOSED SYSTEM

- In this paper, we propose an Adaptive Privacy Policy Prediction (A3P) system which aims to provide users a hassle free privacy settings experience by automatically generating personalized policies. The A3P system handles user uploaded images, and factors in the following criteria that influence one's privacy settings of images:
- The impact of social environment and personal characteristics. Social context of users, such as their profile information and relationships with others may provide useful information regarding users' privacy preferences. For example, users interested in photography may like to share their photos with other amateur photographers.
- The role of image's content and metadata. In general, similar images often incur similar privacy preferences, especially when people appear in the images. For example, one may upload several photos of his kids and specify that only his family members are allowed to see these photos.

c) PROCESS MODEL USED WITH JUSTIFICATION ACCESS CONTROL FOR DATA WHICH REQUIRE USER AUTHENTICATION

The following commands specify access control identifiers and they are typically used to authorize and authenticate the user (command codes are shown in parentheses)

USER NAME (USER)

The user identification is that which is required by the server for access to its file system. This command will normally be the first command transmitted by the user after the control connections are made (some servers may require this).

PASSWORD (PASS)

This command must be immediately preceded by the user name command, and, for some sites, completes the user's identification for access control. Since password information is quite sensitive, it is desirable in general to "mask" it or suppress type out.

FEASIBILITY REPORT

About:

Preliminary investigation examine project feasibility, the likelihood the system will be useful to the organization. The main objective of the feasibility study is to test the Technical, Operational and Economical feasibility for adding new modules and debugging old running system. All system is feasible if they are unlimited resources and infinite time. There are aspects in the feasibility study portion of the preliminary investigation:

- Technical Feasibility
- Operational Feasibility
- Economical Feasibility

A. TECHNICAL FEASIBILITY

The technical issue usually raised during the feasibility stage of the investigation includes the following:

- Does the necessary technology exist to do what is suggested?
- Do the proposed equipment's have the technical capacity to hold the data required to use the new system?

- Will the proposed system provide adequate response to inquiries, regardless of the number or location of users?
- Can the system be upgraded if developed?
- Are there technical guarantees of accuracy, reliability, ease of access and data security?

Earlier no system existed to cater to the needs of 'Secure Infrastructure Implementation System'. The current system developed is technically feasible. It is a web based user interface for audit workflow at NIC-CSD. Thus it provides an easy access to the users.

The database's purpose is to create, establish and maintain a workflow among various entities in order to facilitate all concerned users in their various capacities or roles. Permission to the users would be granted based on the roles specified. Therefore, it provides the technical guarantee of accuracy, reliability and security. The software and hard requirements for the development of this project are not many and are already available in-house at NIC or are available as free as open source.

The work for the project is done with the current equipment and existing software technology. Necessary bandwidth exists for providing a fast feedback to the users irrespective of the number of users using the system.

B. OPERATIONAL FEASIBILITY

Proposed projects are beneficial only if they can be turned out into information system. That will meet the organization's operating requirements. Operational feasibility aspects of the project are to be taken as an important part of the project implementation. Some of the important issues raised are to test the operational feasibility of a project includes the following: -

- Is there sufficient support for the management from the users?
- Will the system be used and work properly if it is being developed and implemented?
- Will there be any resistance from the user that will undermine the possible application benefits?

This system is targeted to be in accordance with the above-mentioned issues. Beforehand, the management issues and user requirements have been taken into consideration. So there is no question of resistance from the users that can undermine the possible application benefits.
The well-planned design would ensure the optimal utilization of the computer resources and would help in the improvement of performance status.

C. ECONOMICAL FEASIBILITY

A system can be developed technically and that will be used if installed must still be a good investment for the organization. In the economical feasibility, the development cost in creating the system is evaluated against the ultimate benefit derived from the new systems. Financial benefits must equal or exceed the costs.

The system is economically feasible. It does not require any addition hardware or software. Since the interface for this system is developed using the existing resources and technologies available at NIC, There is nominal expenditure and economical feasibility for certain.

.

V. LITERATURE SURVEY

A. INTRODUCTION

The project consists of four interrelated thrusts: High availability is one of the key characteristics of Infrastructure-as-a-Service (IaaS) cloud. In this paper, we show a scalable method for availability analysis of large scale IaaS cloud using analytic models. To reduce the complexity of analysis and the solution time, we use an interacting Markov chain based approach. The construction and the solution of the Markov chains is facilitated by the use of a high-level Petri net based paradigm known as stochastic reward net (SRN). Overall solution is composed by iteration over individual SRN sub-model solutions. Dependencies among the sub-models are resolved using fixed-point iteration, for which existence of a solution is proved. We compare the solution obtained from the interacting sub-models with a monolithic model and show that errors introduced by decomposition are insignificant. Additionally, we provide closed form solutions of the sub models and show that our approach can handle very large size IaaS clouds. Cloud computing is a model of Internet-based computing. An IaaS cloud, such as Amazon EC2 and IBM Smart Business Cloud delivers, on-demand, operating system (OS) instances provisioning computational resources in the form of virtual machines deployed in the cloud provider's data center.

Requests submitted by the users are provisioned and served if the cloud has enough available capacity in terms of physical machines. Large cloud service providers such as IBM provide service level agreements (SLAs) regulating the availability of the cloud service. Before committing an SLA to the customers of a cloud, the service provider needs to carry out availability analysis of the infrastructure on which the cloud service is hosted. In this paper, we show how stochastic analytic models can be utilized for cloud service availability analysis.

We first develop a one-level monolithic model. However, such monolithic models become intractable as the size of cloud increases. To overcome this difficulty, we use an interacting sub-models approach. Overall model solution is obtained by iteration over individual sub-model solutions. Comparison of the results with monolithic model shows that errors introduced by model decomposition are negligible. We also develop closed form solutions of the sub-models and show that our approach can scale for large size clouds. To the best of our knowledge, this is the first attempt to analyze availability of a cloud computing infrastructure by using stochastic analytic models.

The presence of three pools of physical machines and the migration of them from one pool to another caused by failure events makes the model both novel and interesting. In order to automate the construction and solution of underlying Markov models, we use a variant of stochastic Petri net (SPN) called stochastic reward net (SRN).

B. HISTORY

A **Web service** is a method of communication between two electronic devices over the World Wide Web. A **Web service** is a software function provided at a network address over the web or the cloud, it is a service that is "always on" as in the concept of utility computing.

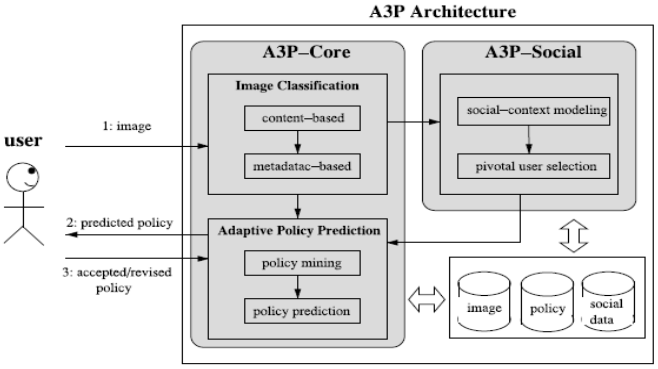
The W3C defines a "Web service" as "a software system designed to support interoperable machine-to-machine interaction over a network". It has an interface described in a machine-processable format (specifically Web Services Description Language, known by the acronym WSDL). Other systems interact with the Web service in a manner prescribed by its description using SOAP messages, typically conveyed using HTTP with an XML serialization in conjunction with other Web-related standards.

The W3C also states, "We can identify two major classes of Web services, REST-compliant Web services, in which the primary purpose of the service is to manipulate XML representations of Web resources using a uniform set of "stateless" operations; and arbitrary Web services, in which the service may expose an arbitrary set of operations

C. PURPOSE

In this article, we propose a two-level framework which according to the user's available history on the site, determines the best available privacy policy for the user's images being uploaded. Our solution relies on an image classification framework for image categories which may be associated with similar policies, and on a policy prediction algorithm to automatically generate a policy for each newly uploaded image, also according to users' social features. Over time, the generated policies will follow the evolution of users' privacy attitude. We provide the results of our extensive evaluation over 5,000 policies, which demonstrate the effectiveness of our system, with prediction accuracies over 90 percent.

VI. ARCHITECTURE



VII. SYSTEM DESIGN

i. INTRODUCTION

Software design sits at the technical kernel of the software engineering process and is applied regardless of the development paradigm and area of application. Design is the first step in the development phase for any engineered product or system. The designer's goal is to produce a model or representation of an entity that will later be built. Beginning, once system requirement have been specified and analyzed, system design is the first of the three technical activities -design, code and test that is required to build and verify software.

The importance can be stated with a single word "Quality". Design is the place where quality is fostered in software development. Design provides us with representations of software that can assess for quality. Design is the only way that we can accurately translate a customer's view into a finished software product or system. Software design serves as a foundation for all the software engineering steps that follow. Without a strong design we risk building an unstable system – one that will be difficult to test, one whose quality cannot be assessed until the last stage.

During design, progressive refinement of data structure, program structure, and procedural details are developed reviewed and documented. System design can be viewed from either technical or project management perspective. From the technical point of view, design is comprised of four activities – architectural design, data structure design, interface design and procedural design.

ii. NORMALIZATION

It is a process of converting a relation to a standard form. The process is used to handle the problems that can arise due to data redundancy i.e. repetition of data in the database, maintain data integrity as well as handling problems that can arise due to insertion, updation, deletion anomalies.

Decomposing is the process of splitting relations into multiple relations to eliminate anomalies and maintain anomalies and maintain data integrity. To do this we use normal forms or rules for structuring relation.

Insertion anomaly: Inability to add data to the database due to absence of other data.
Deletion anomaly: Unintended loss of data due to deletion of other data.
Update anomaly: Data inconsistency resulting from data redundancy and partial update.
Normal Forms: These are the rules for structuring relations that eliminate anomalies.
FIRST NORMAL FORM:

A relation is said to be in first normal form if the values in the relation are atomic for every attribute in the relation. By this we mean simply that no attribute value can be a set of values or, as it is sometimes expressed, a repeating group.

SECOND NORMAL FORM:

A relation is said to be in second Normal form is it is in first normal form and it should satisfy any one of the following rules.

1) Primary key is a not a composite primary key
2) No non key attributes are present
3) Every non key attribute is fully functionally dependent on full set of primary key.

THIRD NORMAL FORM:

A relation is said to be in third normal form if their exits no transitive dependencies.
Transitive Dependency: If two non-key attributes depend on each other as well as on the primary key then they are said to be transitively dependent.

The above normalization principles were applied to decompose the data in multiple tables thereby making the data to be maintained in a consistent state.

iii. E – R DIAGRAMS

- The relation upon the system is structure through a conceptual ER-Diagram, which not only specifics the existential entities but also the standard relations through which the system exists and the cardinalities that are necessary for the system state to continue.
- The entity Relationship Diagram (ERD) depicts the relationship between the data objects.

- The ERD is the notation that is used to conduct the date modeling activity the attributes of each data object noted is the ERD can be described resign a data object descriptions.
- The set of primary components that are identified by the ERD are
  - ◆ Data object ◆ Relationships
  - ◆ Attributes ◆ Various types of indicators.

The primary purpose of the ERD is to represent data objects and their relationships.

iv. DATA FLOW DIAGRAMS

A data flow diagram is graphical tool used to describe and analyze movement of data through a system. These are the central tool and the basis from which the other components are developed. The transformation of data from input to output, through processed, may be described logically and independently of physical components associated with the system. These are known as the logical data flow diagrams. The physical data flow diagrams show the actual implements and movement of data between people, departments and workstations. A full description of a system actually consists of a set of data flow diagrams. Using two familiar notations Yourdon, Gane and Sarson notation develops the data flow diagrams. Each component in a DFD is labeled with a descriptive name. Process is further identified with a number that will be used for identification purpose. The development of DFD'S is done in several levels. Each process in lower level diagrams can be broken down into a more detailed DFD in the next level. The lop-level diagram is often called context diagram. It consists a single process bit, which plays vital role in studying the current system. The process in the context level diagram is exploded into other process at the first level DFD.

The idea behind the explosion of a process into more process is that understanding at one level of detail is exploded into greater detail at the next level. This is done until further explosion is necessary and an adequate amount of detail is described for analyst to understand the process.

Larry Constantine first developed the DFD as a way of expressing system requirements in a graphical from, this lead to the modular design.

A DFD is also known as a "bubble Chart" has the purpose of clarifying system requirements and identifying major transformations that will become programs in system design. So it is the starting point of the design to the lowest level of detail. A DFD consists of a series of bubbles joined by data flows in the system.

v. CONSTRUCTING A DFD:

Several rules of thumb are used in drawing DFD'S:

1. Process should be named and numbered for an easy reference. Each name should be representative of the process.
2. The direction of flow is from top to bottom and from left to right. Data traditionally flow from source to the destination although they may flow back to the source. One way to indicate this is to draw long flow line back to a source. An alternative way is to repeat the source symbol as a destination. Since it is used more than once in the DFD it is marked with a short diagonal.
3. When a process is exploded into lower level details, they are numbered.
4. The names of data stores and destinations are written in capital letters. Process and dataflow names have the first letter of each work capitalized

A DFD typically shows the minimum contents of data store. Each data store should contain all the data elements that flow in and out.

Questionnaires should contain all the data elements that flow in and out. Missing interfaces redundancies and like is then accounted for often through interviews.

vi. SAILENT FEATURES OF DFD'S

1. The DFD shows flow of data, not of control loops and decision are controlled considerations do not appear on a DFD.
2. The DFD does not indicate the time factor involved in any process whether the dataflow take place daily, weekly, monthly or yearly.
3. The sequence of events is not brought out on the DFD.

vii. TYPES OF DATA FLOW DIAGRAMS

1. Current Physical
2. Current Logical
3. New Logical
4. New Physical

1. CURRENT PHYSICAL:

In Current Physical DFD process label include the name of people or their positions or the names of computer systems that might provide some of the overall system-processing label includes an identification of the technology used to process the data. Similarly data flows and data stores are often labels with the names of the actual physical media on which data are stored such as file folders, computer files, business forms or computer tapes.

2. CURRENT LOGICAL:

The physical aspects at the system are removed as much as possible so that the current system is reduced to its essence to the data and the processors that transforms them regardless of actual physical form.

3. NEW LOGICAL:

This is exactly like a current logical model if the user were completely happy with he user were completely happy with the functionality of the current system but had problems with how it was implemented typically through
the new logical model will differ from current logical model while having additional functions, absolute function removal and inefficient flows recognized.

4. NEW PHYSICAL:

The new physical represents only the physical implementation of the new system.

RULES GOVERNING THE DFD'S

PROCESS

1) No process can have only outputs.
2) No process can have only inputs. If an object has only inputs than it must be a sink.
3) A process has a verb phrase label.

DATA STORE

1) Data cannot move directly from one data store to another data store, a process must move data.
2) Data cannot move directly from an outside source to a data store, a process, which receives, must move data from the source and place the data into data store
3) A data store has a noun phrase label.

SOURCE OR SINK

The origin and / or destination of data.

1) Data cannot move direly from a source to sink it must be moved by a process
2) A source and /or sink has a noun phrase land

DATA FLOW

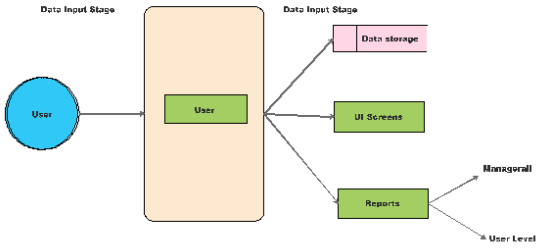
1) A Data Flow has only one direction of flow between symbols. It may flow in both directions between a process and a data store to show a read before an update. The later is usually indicated however by two separate arrows since these happen at different type.
2) A join in DFD means that exactly the same data comes from any of two or more different processes data store or sink to a common location.
3) A data flow cannot go directly back to the same process it leads. There must be atleast one other process that handles the data flow produce some other data flow returns the original data into the beginning process.
4) A Data flow to a data store means update (delete or change).
5) A data Flow from a data store means retrieve or use.

A data flow has a noun phrase label more than one data flow noun phrase can appear on a single arrow as long as all of the flows on the same arrow move together as one package.

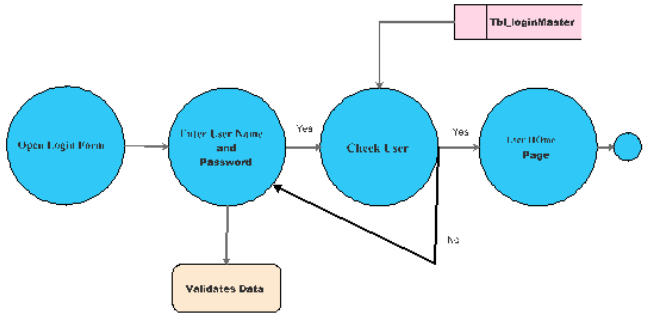
Data Flow Diagrams

DFD Diagrams:

Context Level (0th level DFD)

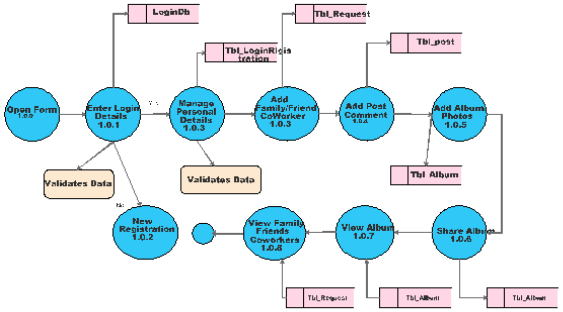


Login DFD



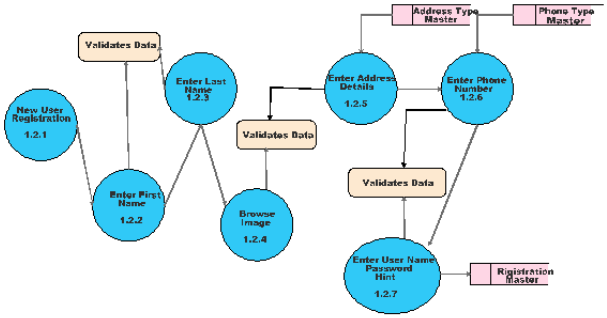
User Functionalities

1st Level



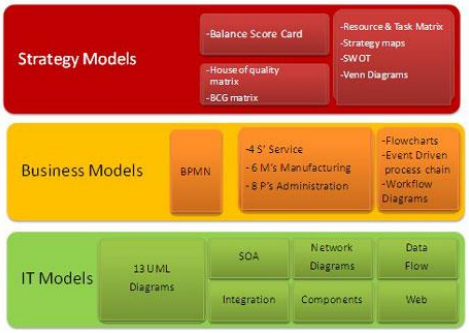
User Functionalities

2nd Level



v. UML DIAGRAMS

The Unified Modeling Language (UML) is used to specify, visualize, modify, construct and document the artifacts of an object-oriented software intensive system under development. The UML uses mostly graphical notations to express the design of software projects. UML offers a standard way to visualize a system's architectural blueprints, including elements such as:

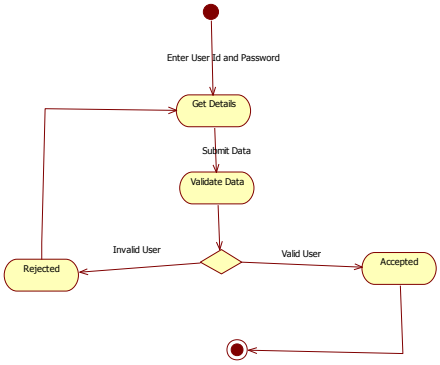


- actors
- business processes
- (logical) components
- activities
- programming language statements
- database schemas, and
- Reusable software components.

DATA DICTONARY

After carefully understanding the requirements of the client the the entire data storage requirements are divided into tables. The below tables are normalized to avoid any anomalies during the course of data entry.

Activity Diagram:



VIII. SYSTEM TESTING AND IMPLEMENTATION

a. INTRODUCTION

Software testing is a critical element of software quality assurance and represents the ultimate review of specification, design and coding. In fact, testing is the one step in the software engineering process that could be viewed as destructive rather than constructive.

A strategy for software testing integrates software test case design methods into a well-planned series of steps that result in the successful construction of software. Testing is the set of activities that can be planned in advance and conducted systematically. The underlying motivation of program testing is to affirm software quality with methods that can economically and effectively apply to both strategic to both large and small-scale systems.

b. STRATEGIC APPROACH TO SOFTWARE TESTING

The software engineering process can be viewed as a spiral. Initially system engineering defines the role of software and leads to software requirement analysis where the information domain, functions, behavior, performance, constraints and validation criteria for software are established. Moving inward along the spiral, we come to design and finally to coding. To develop computer software we spiral in along streamlines that decrease the level of abstraction on each turn.

A strategy for software testing may also be viewed in the context of the spiral. Unit testing begins at the vertex of the spiral and concentrates on each unit of the software as implemented in source code. Testing progress by moving outward along the spiral to integration testing, where the focus is on the design and the construction of the software architecture. Talking another turn on outward on the spiral we encounter validation testing where requirements established as part of software requirements analysis are validated against the software that has been constructed. Finally we arrive at system testing, where the software and other system elements are tested as a whole.

c. UNIT TESTING

Unit testing focuses verification effort on the smallest unit of software design, the module. The unit testing we have is white box oriented and some modules the steps are conducted in parallel.

1. WHITE BOX TESTING

This type of testing ensures that

- All independent paths have been exercised at least once
- All logical decisions have been exercised on their true and false sides
- All loops are executed at their boundaries and within their operational bounds
- All internal data structures have been exercised to assure their validity.

To follow the concept of white box testing we have tested each form .we have created independently to verify that Data flow is correct, All conditions are exercised to check their validity, All loops are executed on their boundaries.

2. BASIC PATH TESTING

Established technique of flow graph with Cyclomatic complexity was used to derive test cases for all the functions. The main steps in deriving test cases were:
Use the design of the code and draw correspondent flow graph.
Determine the Cyclomatic complexity of resultant flow graph, using formula:

V(G)=E-N+2 or
V(G)=P+1 or
V (G) =Number Of Regions
Where V (G) is Cyclomatic complexity,
E is the number of edges,
N is the number of flow graph nodes,
P is the number of predicate nodes.
Determine the basis of set of linearly independent paths.

3. CONDITIONAL TESTING

In this part of the testing each of the conditions were tested to both true and false aspects. And all the resulting paths were tested. So that each path that may be generate on particular condition is traced to uncover any possible errors.

4. DATA FLOW TESTING

This type of testing selects the path of the program according to the location of definition and use of variables. This kind of testing was used only when some local variable were declared. The *definition-use chain* method was used in this type of testing. These were particularly useful in nested statements.

5. LOOP TESTING

In this type of testing all the loops are tested to all the limits possible. The following exercise was adopted for all loops:

- All the loops were tested at their limits, just above them and just below them.

- All the loops were skipped at least once.
- For nested loops test the inner most loop first and then work outwards.
- For concatenated loops the values of dependent loops were set with the help of connected loop.
- Unstructured loops were resolved into nested loops or concatenated loops and tested as above.

IX. SYSTEM SECURITY

i. INTRODUCTION

The protection of computer based resources that include hardware, software, data, procedures and people against unauthorized use or natural Disaster is known as System Security.

System Security can be divided into four related issues:

- Security
- Integrity
- Privacy
- Confidentiality

SYSTEM SECURITY refers to the technical innovations and procedures applied to the hardware and operation systems to protect against deliberate or accidental damage from a defined threat.

DATA SECURITY is the protection of data from loss, disclosure, modification and destruction.

SYSTEM INTEGRITY refers to the power functioning of hardware and programs, appropriate physical security and safety against external threats such as eavesdropping and wiretapping.

PRIVACY defines the rights of the user or organizations to determine what information they are willing to share with or accept from others and how the organization can be protected against unwelcome, unfair or excessive dissemination of information about it.

CONFIDENTIALITY is a special status given to sensitive information in a database to minimize the possible invasion of privacy. It is an attribute of information that characterizes its need for protection.

ii. SECURITY SOFTWARE

It is the technique used for the purpose of converting communication. It transfers message secretly by embedding it into a cover medium with the use of information hiding techniques. It is one of the conventional techniques capable of hiding large secret message in a cover image without introducing many perceptible distortions.

NET has two kinds of security:

- Role Based Security
- Code Access Security

The Common Language Runtime (CLR) allows code to perform only those operations that the code has permission to perform. So CAS is the CLR's security system that enforces security policies by preventing unauthorized access to protected **resources** and **operations**. Using the Code Access Security, you can do the following:

- Restrict what your code can do
- Restrict which code can call your code
- Identify code

CONCLUSION

How to protect users' data privacy is a central question of cloud storage. With more mathematical tools, cryptographic schemes are getting more versatile and often involve multiple keys for a single application. In this article, we consider how to "compress" secret keys in public-key cryptosystems which support delegation of secret keys for different cipher text classes in cloud storage.

No matter which one among the power set of classes, the delegate can always get an aggregate key of constant size. Our approach is more flexible than hierarchical key assignment which can only save spaces if all key-holders share a similar set of privileges. A limitation in our work is the predefined bound of the number of maximum cipher text classes. In cloud storage, the number of cipher texts usually grows rapidly.

So we have to reserve enough cipher text classes for the future extension. Otherwise, we need to expand the public-key as we describe. Although the parameter can be downloaded with cipher texts, it would be better fits size is independent of the maximum number of cipher text classes. On the other hand, when one carries the delegated keys around in a mobile device without using special trusted hardware, the key is prompt to leakage, designing a leakage resilient cryptosystem yet allows efficient and flexible key delegation is also an interesting direction.

BIBLOGRAPHY

[1] A. Acquisti and R. Gross, "Imagined communities: Awareness, information sharing, and privacy on the face book," in Proc. 6th Int. Conf. Privacy Enhancing Technol. Workshop, 2006, pp. 36–58.

[2] R. Agrawal and R. Srikant,"Fast algorithms for mining association rules in large databases," in Proc. 20th Int. Conf. Very Large Data Bases, 1994, pp. 487–499.

[3] S. Ahern, D. Eckles, N. S. Good, S. King, M. Naaman, and R. Nair, "Over exposed?: Privacy patterns and considerations in online and mobile photo sharing," in Proc. Conf. Human Factors Comput. Syst., 2007, pp. 357–366.

[4] M. Ames and M. Naaman, "Why we tag: Motivations for annotation in mobile and online media," in Proc. Conf. Human Factors Comput. Syst., 2007, pp. 971–980.

[5] A. Besmer and H. Lipford, "Tagged photos: Concerns, perceptions, and protections," in Proc. 27th Int. Conf. Extended Abstracts Human Factors Comput. Syst., 2009, pp. 4585–4590.

[6] D. G. Altman and J. M. Bland, "Multiple significance tests: The bonferroni method," Brit. Med. J., vol. 310, no. 6973, 1995.

[7] J. Bonneau, J. Anderson, and L. Church, "Privacy suites: Shared privacy for social networks," in Proc. Symp. Usable Privacy Security, 2009.

[8] J. Bonneau, J. Anderson, and G. Danezis, "Prying data out of a social network," in Proc. Int. Conf. Adv. Soc. Netw. Anal. Mining., 2009, pp.249–254.