Especificación Formal de un Sistema de Control de Acceso para Laboratorios en la Universidad La Salle de Arequipa

Jhordan Huamaní Huamaní[1], Jorge Ortiz Castañeda[1], José Mamani Zuñiga[1], Miguel Flores León[1]
[1]Departamento de Ing. de Software, Universidad La Salle de Arequipa, Perú
{jhordanhh, jortizc, jmamamniz, mfloresl}@ulasalle.edu.pe

***Abstract*—El presente informe aborda la necesidad crítica de implementar un Sistema de Control de Acceso para Laboratorios (SICA-L) en la Universidad La Salle de Arequipa (ULASALLE) para proteger su nueva infraestructura y equipos de alto valor. La ausencia de un sistema formal y robusto representa una vulnerabilidad crítica ante riesgos de seguridad y falta de trazabilidad. El enfoque del proyecto es la especificación formal en VDM++ del SICA-L, ya que los métodos formales son ideales para descubrir ambigüedades y errores en etapas tempranas en sistemas críticos de seguridad. El modelo resultante servirá como un blueprint formal y sin ambigüedades que guiará la implementación del software y facilitará su replicación en otras instituciones de educación superior.**

***Index Terms*—Especificación Formal, Modelos de Control de Acceso, Sistema de Control de Acceso, Invariantes, Verificación de Modelos.**

I. INTRODUCCIÓN

A. Definición del Problema

1) Título del Proyecto: Especificación Formal de un Sistema de Control de Acceso para Laboratorios en la Universidad La Salle de Arequipa (SICA-L ULASALLE).

2) Justificación: La Universidad La Salle de Arequipa está expandiendo su infraestructura académica, dotando a nuevos laboratorios con equipos de alto valor, tecnología de punta y materiales de investigación esenciales. En este contexto, la ausencia de un sistema de control de acceso formal y robusto desde su concepción constituye una vulnerabilidad crítica.

La protección de activos de información e infraestructura física es un requisito indispensable para el funcionamiento de cualquier organización. Los métodos tradicionales, como las bitácoras manuales, son insuficientes para entornos de alto valor debido a que son propensos a errores humanos, falsificación y carecen de capacidades de auditoría en tiempo real.

La falta de un sistema de control de acceso genera riesgos directos significativos:

- **Riesgo de Seguridad:** Facilita el posible robo, daño o mal uso de equipos costosos, comprometiendo la inversión.
- **Riesgo de Integridad Académica:** Permite la manipulación no autorizada de experimentos o datos de investigación.
- **Riesgo de Seguridad Operacional:** El ingreso de personal no capacitado a laboratorios con materiales específicos puede derivar en accidentes.
- **Falta de Trazabilidad:** Impide determinar de manera fehaciente quién se encontraba en las instalaciones durante un incidente, dificultando la rendición de cuentas.

Dado que el sistema aún no existe, la universidad tiene la oportunidad única de diseñar e implementar una solución correcta desde el principio (un desarrollo *greenfield*).

B. Objetivos

1) Objetivo General: Elaborar la especificación formal en VDM++ de un Sistema de Control de Acceso (SICA-L) para los nuevos laboratorios de la Universidad La Salle, que garantice la seguridad y la trazabilidad desde su puesta en marcha y que sirva como un modelo base escalable para otras instituciones de educación superior en la región.

2) Objetivos Específicos:

1) Modelar formalmente las entidades clave del sistema: Usuarios, Laboratorios y los Permisos de acceso que los relacionan.
2) Especificar las políticas y reglas de acceso mediante un método formal para lograr precisión, tal como se requiere para una declaración precisa de las reglas que deben cumplirse.
3) Definir formalmente el conjunto de operaciones críticas (como la verificación de acceso y gestión de permisos) y establecer los invariantes del sistema para asegurar su consistencia lógica, garantizando que el modelo sea robusto frente a estados incoherentes.

II. TRABAJOS RELACIONADOS O ANTECEDENTES

El desarrollo de este proyecto se apoya en varios campos de la seguridad informática y la ingeniería de software:

A. Sistemas de Control de Acceso Físico (PACS)

El sistema propuesto define la lógica que operará sobre diversas tecnologías de hardware, como los identificadores electrónicos que gestionan el acceso a áreas restringidas. En el contexto universitario, la implementación de un sistema de control de acceso basado en tecnología de reconocimiento

facial e inteligencia artificial (IA) es un antecedente directo [1], ya que permite la gestión inteligente de laboratorios, combinando IA y reconocimiento facial para el control de acceso. Este método es considerado más seguro e higiénico en comparación con la identificación por huella dactilar o contraseña.

Modelos de Control de Acceso basados en Roles (RBAC) y Atributos (ABAC) con Dimensiones Temporales: Los roles o permisos de acceso suelen tener una duración temporal o periódica. El modelo Temporal-RBAC (TRBAC) es un antecedente relevante [2], [3], ya que extiende RBAC para soportar la habilitación y deshabilitación periódica de roles y dependencias temporales entre acciones. Las autorizaciones temporales periódicas pueden especificar permisos válidos solo para intervalos específicos, como horarios de 9 a.m. a 1 p.m. en días laborables.

Métodos Formales (VDM++): El uso de VDM++ es crucial [4], ya que permite descubrir ambigüedades y errores en las especificaciones de requisitos en una etapa temprana del ciclo de vida del desarrollo, lo cual es fundamental para sistemas críticos de seguridad. VDM++ proporciona un lenguaje de especificación orientado a objetos con un sólido soporte de herramientas (VDMTools) que permite crear un modelo semántico y ejecutable de las políticas de control de acceso. Esto facilita el análisis y la validación del modelo mediante pruebas (testing).

Especificación Formal de Políticas (XACML): Un antecedente metodológico es la formalización de políticas de control de acceso como XACML (eXtensible Access Control Markup Language) en VDM++ [4]. XACML permite describir políticas sensibles al contexto (context-sensitive), donde la decisión de acceso depende de variables dinámicas del entorno.

Verificación de Políticas de Control de Acceso: La verificación rigurosa de las políticas es un antecedente esencial para garantizar la seguridad. Para los modelos de Control de Acceso Basado en Atributos (ABAC) [5], la verificación de la conformidad del modelo con las políticas y requisitos de seguridad (como la integridad y la cobertura) es crucial para detectar inconsistencias, reglas faltantes o fallas de seguridad como la fuga de privilegios o el bloqueo de acceso autorizado [6].

III. Requerimientos Funcionales (Máximo 5)

El sistema de control de acceso SICA-L debe satisfacer los siguientes requisitos funcionales:

RF1: Gestión de Usuarios: El sistema debe permitir a un administrador registrar, modificar y dar de baja a los usuarios. Cada usuario debe tener un identificador único institucional, nombre completo y un rol definido.

RF2: Gestión de Autorizaciones: El sistema debe permitir a un administrador asignar y revocar el permiso de acceso de un usuario a uno o más laboratorios específicos.

RF3: Verificación de Acceso: El sistema debe proveer una operación para procesar una solicitud de acceso, verificando la identidad del usuario y su autorización vigente para el laboratorio en cuestión.

RF4: Registro de Eventos (Bitácora): El sistema debe registrar de forma inmutable cada intento de acceso. Este registro de auditoría (*audit trail*) es esencial para la seguridad, ya que proporciona los medios para detectar y analizar las violaciones de la política. La bitácora debe almacenar ID de usuario, laboratorio, fecha, hora y resultado.

RF5: Registro de Ingreso y Salida: Tras un acceso aprobado, el sistema debe registrar formalmente el ingreso de un usuario al laboratorio. Debe existir una operación complementaria para registrar su salida.

IV. Metodología y Desarrollo

La metodología de desarrollo se centrará en la Especificación Formal (VDM++) para asegurar la corrección y robustez del modelo de acceso antes de su implementación.

1) **Modelado de Entidades y Tipos de Datos:** El primer paso es la traducción de los conceptos del mundo real a un modelo matemático, aprovechando la separación de tipos de datos y funcionalidad de VDM++. Se definirán las clases (*clases*) y variables de instancia (*instance variables*) para las entidades clave (Usuarios, Laboratorios, Permisos, Roles). Se utilizarán tipos abstractos como mapas, conjuntos y registros para describir el estado local del sistema.
2) **Definición de Invariantes y Operaciones Críticas:** Se definirán formalmente las operaciones que rigen el sistema (RF1, RF2, RF3, RF5) mediante precondiciones y postcondiciones. Los invariantes del sistema se establecerán para asegurar la consistencia lógica, garantizando que el modelo no pueda alcanzar un estado incoherente tras la ejecución de las operaciones.
3) **Especificación de Políticas de Control de Acceso:** Las reglas de acceso se especificarán utilizando un lenguaje de lógica formal, similar a la estructura de reglas de XACML, donde las decisiones de acceso pueden depender de atributos del sujeto, el objeto y las condiciones ambientales (tiempo, rol, etc.). Esto se modela mediante expresiones lógicas (*FExp* en VDM++) que se evalúan en función del entorno dinámico.
4) **Verificación y Validación del Modelo:** El modelo formal se someterá a validación mediante la ejecución directa en el intérprete de VDMTools. Se aplicarán pruebas (*testing*) utilizando peticiones de acceso (*requests*) simuladas para obtener retroalimentación rápida sobre el comportamiento de la política. Para problemas más complejos (como la verificación de la lógica de las reglas ABAC), pueden emplearse métodos de *pseudo-exhaustive testing* que utilizan técnicas combinatorias (como ACTS) para generar conjuntos de pruebas eficientes que aseguren la cobertura de las interacciones

de atributos y condiciones, reduciendo el tamaño del conjunto de pruebas a un nivel manejable. Esto es crucial para detectar fallos semánticos como reglas conflictivas o incompletas.

V. Resultados y Discusión

El resultado principal del proyecto es la generación de un modelo de especificación formal y ejecutable en VDM++.

```
class SistemaControlAcceso

types
  public Rol = <ADMIN> | <DOCENTE> | <ESTUDIANTE> | <INVESTIGADOR>;

instance variables
  usuarios : set of Usuario := {};
  laboratorios : set of Laboratorio := {};
  permisos : map nat1 to set of nat1 := {|->};
  bitacora : seq of Evento := [];

inv
  card usuarios = card {u.id | u in set usuarios};
inv
  card laboratorios = card {l.id | l in set laboratorios};
inv
  forall uid in set dom permisos &
    uid in set {u.id | u in set usuarios} and
    forall lid in set permisos(uid) &
      lid in set {l.id | l in set laboratorios};
inv
  forall e in set elems bitacora &
    e.usuario in set {u.id | u in set usuarios} and
    e.laboratorio in set {l.id | l in set laboratorios};
inv
  dom permisos subset {u.id | u in set usuarios};

operations

  public RegistrarUsuario : Usuario ==> bool
  RegistrarUsuario(u) ==
  (
    if u.id in set {x.id | x in set usuarios} then
      return false
    else
    (
      usuarios := usuarios union {u};
      return true;
    )
  );

  public ModificarUsuario : nat1 * seq of char * Rol ==> bool
  ModificarUsuario(uid, nombre, rol) ==
  (
    dcl uSet : set of Usuario := {x | x in set usuarios & x.id = uid};
    if card uSet = 0 then
      return false
    else
    (
      let u in set uSet in
        usuarios := (usuarios \ {u}) union
                    {new Usuario(uid, nombre, rol)};
      return true
    )
  );
```

Fig. 1. Especificación de la Clase Sistema Parte 1 en VDM++. (Fuente: Archivo SistemaControlAcceso.vdmpp)

A. Discusión sobre Seguridad y Precisión Formal

La aplicación de métodos formales garantiza que la política de acceso será una declaración precisa de las reglas que deben cumplirse. La especificación formal permite identificar discrepancias entre los requisitos de alto nivel y la implementación deseada, previniendo vulnerabilidades serias en el mecanismo de control de acceso.

La especificación del SICA-L debe apuntar a la seguridad del sistema, la trazabilidad y la resistencia a ataques comunes. Dado que el sistema requiere una autenticación previa para el control de acceso (RF3), la elección de la tecnología de autenticación es relevante. Tecnologías como ECC (*Elliptic Curve Cryptography*) basadas en RFID son antecedentes sólidos en entornos de IoT de alta sensibilidad (como la atención médica) [7], ofreciendo costos computacionales y de comunicación más bajos y resistencia probada contra ataques de *Man-in-the-Middle*, *Replay attack* y ataques de falsificación (*forging attack*). Implementar un protocolo de autenticación mutua (donde la tarjeta autentica al lector y viceversa) es una característica de seguridad clave que resiste ataques de Denegación de Servicio (DoS).

B. Discusión sobre la Complejidad del Diseño

Aunque el uso de métodos formales garantiza precisión y coherencia lógica (a través de invariantes), un modelo de control de acceso preciso puede ser inherentemente complejo. Los modelos detallados de control de acceso que involucran atributos o roles temporales (como TRBAC) pueden llevar a especificaciones que, si no se verifican, pueden ser ambiguas o inconsistentes. El proceso de verificación del modelo se vuelve tan crucial como la modelización misma, enfocándose en la verificación de la integridad, la cobertura y la ausencia de fallas como la fuga de privilegios (*privilege leakage*) o el bloqueo de privilegios (*privilege blocking*).

```
class Evento

instance variables
  public usuario : nat1;
  public laboratorio : nat1;
  public fechaHora : seq of char;
  public tipo : <INTENTO_ACCESO> | <INGRESO> | <SALIDA>;
  public resultado : bool;

inv usuario > 0;
inv laboratorio > 0;
inv len fechaHora > 0;

operations
  public Evento : nat1 * nat1 * seq of char * (<INTENTO_ACCESO> | <INGRESO> | <SALIDA>) * bool ==> Evento
  Evento(u, l, f, t, r) ==
  (
    usuario := u;
    laboratorio := l;
    fechaHora := f;
    tipo := t;
    resultado := r;
    return self;
  );

end Evento
```

Fig. 2. Especificación de la Clase Evento en VDM++. (Fuente: Archivo Evento.vdmpp)

```
class Laboratorio

instance variables
  public id : nat1;
  public nombre : seq of char;
  public ubicacion : seq of char;

inv id > 0;
inv len nombre > 0;
inv len ubicacion > 0;

operations
  public Laboratorio : nat1 * seq of char * seq of char ==> Laboratorio
  Laboratorio(i, n, u) ==
  (
    id := i;
    nombre := n;
    ubicacion := u;
    return self;
  );

end Laboratorio
```

Fig. 3. Especificación de la Clase Laboratorio en VDM++. (Fuente: Archivo Laboratorio.vdmpp)

VI. Conclusiones

La especificación formal de un Sistema de Control de Acceso para Laboratorios en la Universidad La Salle de Arequipa (SICA-L) es una respuesta necesaria y proactiva a la vulnerabilidad de la nueva infraestructura ante la falta de un mecanismo robusto y auditable. El proyecto busca elaborar un modelo en VDM++ que defina con precisión las entidades,

```
class Usuario
types
  public Rol = <ADMIN> | <DOCENTE> | <ESTUDIANTE> | <INVESTIGADOR>;
instance variables
  public id : nat1;
  public nombre : seq of char;
  public rol : Rol;

inv id > 0;
inv len nombre > 0;

operations
  public Usuario : nat1 * seq of char * Rol ==> Usuario
  Usuario(i, n, r) ==
  (
    id := i;
    nombre := n;
    rol := r;
    return self;
  );

end Usuario
```

Fig. 4. Especificación de la Clase Usuario en VDM++. (Fuente: Archivo Usuario.vdmpp)

```
  public DarBajaUsuario : nat1 ==> bool
  DarBajaUsuario(uid) ==
  (
    dcl existentes : set of Usuario := {u | u in set usuarios & u.id = uid};
    if card existentes = 0 then
      return false
    else
    (
      usuarios := usuarios \ existentes;
      if uid in set dom permisos then
        permisos := {uid} <-: permisos;
      return true
    )
  );

  public RegistrarLaboratorio : Laboratorio ==> bool
  RegistrarLaboratorio(l) ==
  (
    if l.id in set {x.id | x in set laboratorios} then
      return false
    else
    (
      laboratorios := laboratorios union {l};
      return true;
    )
  );

  public ModificarLaboratorio : nat1 * seq of char * seq of char ==> bool
  ModificarLaboratorio(lid, nombre, ubicacion) ==
  (
    dcl lSet : set of Laboratorio := {x | x in set laboratorios & x.id = lid};
    if card lSet = 0 then
      return false
    else
    (
      let l in set lSet in
        laboratorios := (laboratorios \ {l}) union
            {new Laboratorio(lid, nombre, ubicacion)};
      return true
    )
  );

  public DarBajaLaboratorio : nat1 ==> bool
  DarBajaLaboratorio(lid) ==
  (
    dcl existentes : set of Laboratorio := {l | l in set laboratorios & l.id = lid};
    if card existentes = 0 then
      return false
    else
    (
      laboratorios := laboratorios \ existentes;
      for all uid in set dom permisos do
        permisos(uid) := permisos(uid) \ {lid};
      return true;
    )
  );
```

Fig. 5. Especificación de la Clase Sistema Parte 2 en VDM++. (Fuente: Archivo SistemaControlAcceso.vdmpp)

las operaciones críticas, y las reglas de acceso, asegurando la consistencia lógica mediante la formulación de invariantes. La metodología, basada en la ejecutabilidad de VDM++ y las técnicas de verificación de modelos (como las pruebas combinatorias y el *model checking*), asegura que el *blueprint* resultante está libre de las ambigüedades y fallos que son comunes en las especificaciones de requisitos de seguridad tradicionales.

Al establecer un modelo base escalable para ULASALLE y

```
  public AsignarPermiso : nat1 * nat1 ==> bool
  AsignarPermiso(uid, lid) ==
  (
    if not (uid in set {u.id | u in set usuarios}) then
      return false
    elseif not (lid in set {l.id | l in set laboratorios}) then
      return false
    else
    (
      if uid in set dom permisos then
        permisos(uid) := permisos(uid) union {lid}
      else
        permisos := permisos ++ {uid |-> {lid}};
      return true;
    )
  );

  public RevocarPermiso : nat1 * nat1 ==> bool
  RevocarPermiso(uid, lid) ==
  (
    if uid not in set dom permisos then
      return false
    elseif lid not in set permisos(uid) then
      return false
    else
    (
      permisos(uid) := permisos(uid) \ {lid};
      return true;
    )
  );

  public VerificarAcceso : nat1 * nat1 ==> bool
  VerificarAcceso(uid, lid) ==
  (
    return (uid in set dom permisos and lid in set permisos(uid));
  );

  public RegistrarIntentoAcceso : nat1 * nat1 * bool ==> bool
  RegistrarIntentoAcceso(uid, lid, resultado) ==
  (
    if uid not in set {u.id | u in set usuarios} or
       lid not in set {l.id | l in set laboratorios} then
      return false
    else
    (
      dcl e : Evento := new Evento(uid, lid, "2025-10-21 00:00", <INTENTO_ACCESO>, resultado);
      bitacora := bitacora ^ [e];
      return true;
    )
  );
```

Fig. 6. Especificación de la Clase Sistema Parte 3 en VDM++. (Fuente: Archivo SistemaControlAcceso.vdmpp)

```
  public RegistrarIngreso : nat1 * nat1 ==> bool
  RegistrarIngreso(uid, lid) ==
  (
    if not VerificarAcceso(uid, lid) then
      return false
    else
    (
      dcl e : Evento := new Evento(uid, lid, "2025-10-21 00:00", <INGRESO>, true);
      bitacora := bitacora ^ [e];
      return true
    )
  );

  public RegistrarSalida : nat1 * nat1 ==> bool
  RegistrarSalida(uid, lid) ==
  (
    if uid not in set {u.id | u in set usuarios} or
       lid not in set {l.id | l in set laboratorios} then
      return false
    else
    (
      dcl e : Evento := new Evento(uid, lid, "2025-10-21 00:00", <SALIDA>, true);
      bitacora := bitacora ^ [e];
      return true;
    )
  );

  public HistorialLaboratorio : nat1 * set of (<INTENTO_ACCESO> | <INGRESO> | <SALIDA>) ==> seq of Evento
  HistorialLaboratorio(lid, tipos) ==
  (
    dcl filtrado : seq of Evento := [];
    for all e in set elems bitacora do
      if e.laboratorio = lid and e.tipo in set tipos then
        filtrado := filtrado ^ [e];
    return filtrado
  );

  public HistorialUsuario : nat1 * set of (<INTENTO_ACCESO> | <INGRESO> | <SALIDA>) ==> seq of Evento
  HistorialUsuario(uid, tipos) ==
  (
    dcl filtrado : seq of Evento := [];
    for all e in set elems bitacora do
      if e.usuario = uid and e.tipo in set tipos then
        filtrado := filtrado ^ [e];
    return filtrado
  );

end SistemaControlAcceso
```

Fig. 7. Especificación de la Clase Sistema Parte 4 en VDM++. (Fuente: Archivo SistemaControlAcceso.vdmpp)

potencialmente para otras instituciones de educación superior, el proyecto contribuye a elevar el estándar de la seguridad física y la trazabilidad en entornos académicos de alto valor.

REFERENCIAS

[1] Z. Qin, J. Guo, L. Huang, Y. Huang, and C. Leng, "Design and implementation of face recognition access control system for university laboratory based on artificial intelligence technology," in *Proceedings of the 2nd International Conference on Internet, Education and Information Technology (IEIT 2022)*, 2022, pp. 250–254. [Online]. Available: https://doi.org/10.2991/978-94-6463-058-9_41

[2] E. Bertino, P. A. Bonatti, and E. Ferrari, "Trbac: A temporal role-based access control model," *ACM Transactions on Information and System Security*, vol. 4, no. 3, pp. 191–233, Aug. 2001. [Online]. Available: https://dl.acm.org/doi/10.1145/501978.501979

[3] E. Bertino, C. Bettini, E. Ferrari, and P. Samarati, "An access control model supporting periodicity constraints and temporal reasoning," *ACM Transactions on Information and System Security*, vol. 2, no. 3, pp. 231–285, Aug. 1999. [Online]. Available: https://doi.org/10.1145/293910.293151

[4] J. W. Bryans and J. S. Fitzgerald, "Formal engineering of xacml access control policies in vdm++," in *International Conference on Formal Engineering Methods (ICFEM 2007)*, ser. Lecture Notes in Computer Science, vol. 4789. Springer, Nov. 2007, pp. 37–56. [Online]. Available: https://doi.org/10.1007/978-3-540-76650-6_4

[5] V. C. Hu, D. Ferraiolo, R. Kuhn, A. Schnitzer, K. Sandlin, R. Miller, and K. Scarfone, "Guide to attribute based access control (abac) definition and considerations," National Institute of Standards and Technology (NIST), Tech. Rep. Special Publication 800-162, Jan. 2014. [Online]. Available: https://doi.org/10.6028/NIST.SP.800-162

[6] V. C. Hu, R. Kuhn, and D. Yaga, "Verification and test methods for access control policies/models," National Institute of Standards and Technology (NIST), Tech. Rep. Special Publication 800-192, Jun. 2017. [Online]. Available: https://doi.org/10.6028/NIST.SP.800-192

[7] D. Noori, H. Shakeri, and M. N. Torshiz, "An elliptic curve cryptosystem-based secure rfid mutual authentication for internet of things in healthcare environment," *EURASIP Journal on Wireless Communications and Networking*, vol. 2022, no. 1, p. 64, 2022. [Online]. Available: https://doi.org/10.1186/s13638-022-02146-y