Learning on the Move: A Pedagogical Framework for State-of-the-Art Mobile Learning

- Dispositivos móviles están contribuyendo a nivelar las condiciones para personas de diferentes razas y orígenes socioeconómicos.
- El aprendizaje basado en juegos se refiere al uso y desarrollo de mecánicas de juego en contextos "no lúdicos".
- Correlación entre la motivación al estudio y la retención. Hay 6 factores que se combinan para crear un estado de motivación intrínseca: desafío, control, fantasía, competencia, cooperación y reconocimiento.

Tabla 1: Tendencias en el aprendizaje móvil

1	Flash a HTML5	Fácil de implementar
2	Análisis de aplicaciones móviles	Comprensión de la interacción de los alumnos y comportamiento
3	Desarrollo web responsivo para consistencia multidispositivo	Multiplataforma
4	Los móviles como dispositivos para aprendizaje basado en competencias	Tendencia de videos móviles y simuladores para profesionales (industria)
5	Aprendizaje sensible a la ubicación geográfica	Modelos personalizados (alumnos) para monitorear progreso y acceder a contenido (expertos) según su geolocalización
6	Aprendizaje móvil social	Compartir conocimiento e impulsar innovación
7	Enfoque de diseño independiente del dispositivo	BYOD
8	Uso multipantalla	Pasar de una pantalla a otra (NORMA)
9	Aprendizaje del tamaño de un byte para el rendimiento apoyo	Móviles como medio de soporte (información digerible)
10	Aprendizaje y asistencia gamificados	Gamificación (mecánicas de juegos no lúdicos)
11	Realidad aumentada para móviles	Apps móviles
12	Dispositivos portátiles	Monitoreo

Tabla 2: Consejos de diseño para el aprendizaje móvil

1	Minimizar las funcionalidades de la GUI	na
2	Dividir el curso en varios módulos	Módulos cortos captan mejor la atención

3	Ofrece nuggets pequeños	na
4	Reemplazar texto largo por audio	na
5	Evite gráficos e imágenes de fondo complicados	Distracción
6	Minimizar scrolls	Usar desplazamientos verticales (móvil)
7	Diseño para un uso con una sola mano	na
8	Interactividades sencillas	Images, Icons, etc.

Tabla3: El futuro del aprendizaje

Un vistazo al futuro del aprendizaje	
Cambios a un ecosistema donde el aprendizaje se adapte al niño	Aprendizaje no definido por tiempo y lugar
	Personalización radical será común
	Datos sugerirán estrategias educativas
	Credenciales reflejarán aprendizaje diverso
	Comunidades controlarán el aprendizaje
	Innovación social resolverá recursos
	Roles de aprendizaje serán variados
	Redes digitales conectarán para aprender
	Escuela será auto-organizada

	Preparación continua para carreras
	Listas de aprendizaje personalizadas
	Experiencias digitales y locales

Software Construction Technologies

Tecnología	Descripción	Características
Maven	Es una herramienta de gestión y comprensión de proyectos Java.	Gestión de dependencias: Automatiza la gestión de dependencias de proyectos. Lifecycles de proyectos: Define fases del ciclo de vida del proyecto. Plugins: Soporte para plugins que amplían su funcionalidad. Repositorios de artefactos: Gestiona repositorios de artefactos. Integración continua: Facilita la integración continua y despliegue continuo.
Gradle	Es un sistema de automatización de construcción que compila, prueba, publica y empaqueta proyectos de software.	DSL basado en Groovy: Utiliza un lenguaje de dominio específico basado en Groovy. Configuración flexible: Permite configuraciones detalladas y personalizables. Incremental builds: Solo reconstruye lo necesario.

Tecnología	Descripción	Características
		Multi-project builds: Soporte para múltiples proyectos en un solo script. Integración con herramientas: Se integra con herramientas como Jenkins y Docker.
CMake	Es una herramienta de construcción que controla el proceso de compilación usando archivos de configuración.	Multiplataforma: Funciona en diferentes sistemas operativos. Generación de Makefiles: Genera Makefiles o proyectos de construcción. Configuración basada en scripts: Configuración a través de scripts CMakeLists.txt. Mapeo de dependencias: Gestiona dependencias entre archivos. Soporte para IDEs: Soporte para varios entornos de desarrollo integrados.
GitHub + GitLab	Son plataformas de control de versiones y colaboración.	Control de versiones: Gestiona versiones de código fuente. Colaboración: Facilita la colaboración entre desarrolladores. Pull requests: Permite revisiones de código y fusiones. Issue tracking: Gestiona problemas y tareas. Integración continua: Soporte

Tecnología	Descripción	Características
		para integración continua y despliegue continuo.
Jenkins, Travis CI, CircleCI	Son herramientas de integración continua.	Automatización de pruebas: Automatiza pruebas de software. Integración continua: Facilita la integración continua. Despliegue continuo: Soporte para despliegue continuo. Plugins: Amplia gama de plugins para extender funcionalidad. Multi-branch builds: Soporte para múltiples ramas de desarrollo.
Docker	Es una plataforma para desarrollar, enviar y ejecutar aplicaciones en contenedores.	Contenedores: Crea y gestiona contenedores de aplicaciones. Portabilidad: Asegura la portabilidad de aplicaciones. Isolación: Aísla aplicaciones y sus dependencias. Escala: Facilita la escalabilidad de aplicaciones. Orquestación: Soporte para herramientas de orquestación como Kubernetes.
Mercurial	Es una herramienta de control de versiones distribuido.	Control de versiones distribuido: Gestiona versiones de código fuente. Historial de cambios: Mantiene un historial detallado de cambios.

Tecnología	Descripción	Características
		Revisión de código: Facilita la revisión de código. Integración con herramientas: Se integra con herramientas de desarrollo. Compatibilidad con Git: Soporte para trabajar con repositorios Git.

La Tecnología RFID

Resumen detallado del documento sobre Tecnología RFID:

El documento proporciona una revisión exhaustiva sobre la **tecnología RFID (Identificación por Radiofrecuencia)**, enfocándose en su funcionamiento, aplicaciones y los desafíos actuales que enfrenta.

1. Introducción a la Tecnología RFID:

La tecnología RFID utiliza ondas de radio para identificar y rastrear automáticamente objetos. Los sistemas RFID constan de tres componentes clave: una etiqueta (tag), un lector (reader) y una antena que facilita la comunicación. Las etiquetas RFID pueden ser activas, pasivas o semipasivas dependiendo de si tienen su propia fuente de energía o dependen del lector para activarse.

2. Funcionamiento del RFID:

- **Etiquetas RFID**: Las etiquetas contienen un chip de silicio y una antena que les permite comunicarse con los lectores RFID mediante señales de radio. Los datos almacenados en estas etiquetas pueden ser únicos, lo que facilita el seguimiento y la identificación.
- **Lectores RFID**: Estos dispositivos emiten señales de radio a las etiquetas para activar su respuesta. Pueden leer múltiples etiquetas a la vez, y la distancia de lectura varía según la frecuencia utilizada.

3. Tipos de RFID según Frecuencia:

El documento detalla las distintas bandas de frecuencia que se utilizan en los sistemas RFID:

- **Baja frecuencia (LF)**: Operan entre 30 kHz y 300 kHz, tienen un rango de lectura corto y se utilizan para aplicaciones como el control de acceso.
- **Alta frecuencia (HF)**: Operan alrededor de 13.56 MHz, con aplicaciones comunes como el pago sin contacto y las tarjetas inteligentes.
- **Ultra alta frecuencia (UHF)**: Con un rango mayor, entre 300 MHz y 3 GHz, se utilizan en la logística y la gestión de inventarios.

4. Aplicaciones del RFID:

El documento abarca diversas aplicaciones de RFID en distintos sectores:

- **Logística y cadena de suministro**: Mejora la eficiencia en el seguimiento de productos, desde el almacenamiento hasta la entrega.
- **Seguridad**: Utilizada en tarjetas de acceso y sistemas de autenticación de personal.
- **Salud**: Seguimiento de equipos médicos y administración de medicamentos en hospitales.
- **Comercio**: RFID se usa en inventarios y pagos sin contacto.

5. Beneficios del RFID:

- **Automatización**: Elimina la necesidad de intervención manual en procesos de identificación.
- **Precisión y eficiencia**: Puede escanear múltiples etiquetas simultáneamente, reduciendo errores.
- **Mejora de la seguridad**: Proporciona un mayor control y seguimiento de los activos.

6. Desafíos y Consideraciones:

El documento también menciona los desafíos tecnológicos de la implementación de RFID:

- **Interferencia de señales**: Las ondas de radio pueden ser afectadas por obstáculos físicos o interferencias de otros dispositivos.
- **Costo**: Aunque el precio de las etiquetas ha disminuido, sigue siendo un factor importante, especialmente en sistemas de gran escala.
- **Privacidad y seguridad**: La información transmitida por las etiquetas RFID puede ser interceptada, lo que plantea preocupaciones sobre la privacidad y el uso indebido de datos.

7. Soluciones a los Desafíos:

Se proponen soluciones como el uso de protocolos de cifrado para proteger la información y el desarrollo de materiales que minimicen las interferencias de señal.

8. Futuro de RFID:

El documento concluye con una visión del futuro de RFID, donde se espera que la tecnología continúe expandiéndose en nuevas áreas como el Internet de las Cosas (IoT), ciudades inteligentes y la automatización industrial.

Cryptanalysis of the David-Prasad RFID Ultralightweight Authentication Protocol

El documento **"Cryptanalysis of the David-Prasad RFID Ultralightweight Authentication Protocol"** aborda un análisis criptográfico del protocolo propuesto por David y Prasad en 2009 para la autenticación mutua de etiquetas RFID de bajo costo. A continuación, se ofrece un resumen detallado de las secciones 3, 4 y 5:

3. Ataque de Rastreabilidad

La rastreabilidad es un problema crítico en los sistemas RFID, ya que compromete la privacidad al permitir que los movimientos de las etiquetas sean seguidos. El protocolo

David-Prasad utiliza pseudónimos en lugar de identificadores estáticos, pero el ataque presentado en esta sección demuestra que el protocolo no garantiza un nivel adecuado de anonimato.

Descripción del ataque:

1. **Modelo de adversario:** El adversario (A) tiene la capacidad de espiar pasivamente las comunicaciones entre el lector (R) y la etiqueta (T).
2. **Juego de rastreabilidad:** Se estructura en tres fases:
    - **Fase de aprendizaje:** El adversario observa una ejecución genuina del protocolo y obtiene el pseudónimo y los mensajes intercambiados.
    - **Fase de desafío:** A elige dos etiquetas, T0 y T1, y recibe los mensajes correspondientes a uno de los dos (seleccionado al azar) sin saber cuál. El objetivo es determinar si los mensajes corresponden a T0 o T1.
    - **Fase de adivinanza:** El adversario adivina cuál de las dos etiquetas generó los mensajes observados.

El ataque se basa en calcular la probabilidad de que un adversario pueda distinguir entre los pseudónimos de dos etiquetas y deducir la identidad de la etiqueta observada. Se demuestra que el protocolo no es resistente a este tipo de ataque, ya que un adversario pasivo puede rastrear una etiqueta con un 62,5% de precisión tras espiar un solo intercambio de mensajes.

4. Fuga de Secretos Almacenados

Además de los problemas de rastreabilidad, el protocolo también expone a las etiquetas a la fuga de sus secretos a largo plazo, como el identificador estático (ID) y las claves secretas (K1 y K2). La estructura del ataque es similar a la anterior, pero ahora el adversario utiliza las respuestas de múltiples sesiones consecutivas para filtrar los bits del identificador estático y las claves.

Mecánica del ataque:

- Al observar dos sesiones consecutivas, el adversario puede calcular la diferencia entre los pseudónimos antiguos y nuevos, lo que revela información sobre los números aleatorios generados en cada sesión.
- Combinando esta información con las operaciones XOR y AND realizadas por el protocolo, el adversario puede filtrar gradualmente los bits de K1 y K2. En

promedio, un adversario puede recuperar un cuarto de los bits de las claves en solo dos sesiones.

Este ataque tiene consecuencias devastadoras, ya que permite al adversario recuperar por completo la clave secreta de la etiqueta y su identificador estático, comprometiendo toda la seguridad del sistema.

5. Criptoanálisis Pasivo Tango

Esta sección presenta un ataque novedoso llamado "Tango", que es extremadamente eficiente y permite a un atacante pasivo recuperar todas las claves secretas y el identificador estático de una etiqueta. Se divide en dos fases:

1. **Selección de buenas aproximaciones:** El atacante identifica combinaciones simples de los mensajes intercambiados que se correlacionan fuertemente con los valores secretos (K1, K2, ID). Estas combinaciones se basan en distancias de Hamming y se seleccionan aquellas que se desvían significativamente del valor esperado.
2. **Combinación de aproximaciones:** Tras varias sesiones espiadas, el atacante combina las aproximaciones obtenidas para formar una estimación precisa de los valores secretos. Este proceso involucra el almacenamiento de las aproximaciones en matrices y la aplicación de una función de umbral que determina los bits más probables en función de la frecuencia de ocurrencia en cada columna.

Resultados:

- Después de espiar aproximadamente 65 sesiones, el atacante puede recuperar por completo las claves K1 y K2.
- Para el identificador estático, se necesitan alrededor de 50 sesiones para recuperar los 96 bits del identificador, aunque el 90% de los bits pueden recuperarse tras solo 10 sesiones, lo que reduce significativamente el esfuerzo necesario.

Este ataque es devastador para la seguridad del protocolo, ya que permite a un adversario pasivo recuperar toda la información secreta sin necesidad de interactuar directamente con las etiquetas. Además, el atacante podría clonar o suplantar las etiquetas y romper su sincronización con el servidor.

The Cost of Poor Quality Software in the US: A 2018 Report

Es un análisis detallado del impacto financiero y organizacional del software de baja calidad en los Estados Unidos, destacando que los costos asociados a este problema en 2018 alcanzaron los 2.84 billones de dólares, sin contar el impacto de la deuda técnica. Este reporte está dirigido principalmente a ejecutivos de nivel C, como CTOs y CIOs, y se basa en fuentes públicas que evalúan cómo los problemas de calidad en el software afectan la economía y la industria en general.

Resumen detallado del contenido:

1. **Resumen Ejecutivo**:
    - La investigación revela una carencia generalizada en las organizaciones de TI respecto a la medición del costo de la calidad del software (CoSQ), lo que limita la capacidad de los líderes para cuantificar los costos asociados a la baja calidad del software.
    - Se introduce el modelo "iceberg" para visualizar los costos ocultos relacionados con la mala calidad del software, que se hacen visibles solo después de fallos críticos.
2. **Contexto y Motivaciones**:
    - El software es esencial para el funcionamiento moderno de la sociedad, y su mala calidad puede generar problemas graves, incluyendo proyectos cancelados, defectos críticos y aumento de la deuda técnica.
    - En 2018, los costos del software defectuoso incluyen deuda técnica (18%), fallos de software (37%), problemas con sistemas heredados (21%) y proyectos cancelados o problemáticos (6%).
3. **Paisaje Tecnológico**:
    - **Mirando hacia atrás**: Los sistemas heredados son una gran fuente de gastos, consumiendo hasta el 75% de los presupuestos de TI.
    - **Mirando al futuro**: La rápida innovación tecnológica aumenta la vulnerabilidad de los sistemas y la deuda técnica.
    - **Situación actual**: Los sistemas interconectados y complejos de hoy en día son propensos a fallos masivos y defectos críticos.

4. **Perspectiva del Talento Humano**:
    - El informe analiza la escasez de profesionales de TI calificados, lo que aumenta el costo del software de baja calidad debido a la alta demanda y falta de capacitación adecuada.
    - El impacto de la economía "gig" en el sector de TI también afecta la consistencia y calidad de los productos de software.
5. **Costos de Calidad del Software**:
    - El costo de la mala calidad del software (CPSQ) abarca varios aspectos: deuda técnica, defectos, sistemas problemáticos y cancelación de proyectos.
    - El estudio concluye que el CPSQ en los EE. UU. en 2018 asciende a aproximadamente 2.84 billones de dólares.
6. **Conclusiones y Recomendaciones**:
    - Se enfatiza la necesidad de invertir en medidas proactivas como la detección temprana de problemas y la adopción de prácticas de desarrollo de software más rigurosas para mitigar los costos de la baja calidad del software.
    - El reporte recomienda que las organizaciones midan su CPSQ y enfoquen sus esfuerzos en prevenir fallos y mejorar continuamente la calidad del software.