



BEHAVIORAL CODE ANALYSIS

SEMINARIO 3 - EQUIPO 10

ÁNGELA ROZA MORENO - UO284342

CARLOS SANCHEZ RODRIGUEZ - UO282621

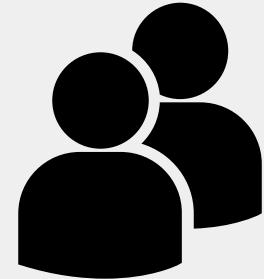
RODRIGO GARCÍA IGLESIAS - UO276396

SANTIAGO LÓPEZ LASO - UO277369

ÍNDICE

- | | | | |
|-----------|--|-----------|---|
| 01 | ¿QUE ES? | 08 | PROYECTOS GREENFIELD |
| 02 | ¿QUÉ PROBLEMAS AYUDA A RESOLVER? | 09 | INTEGRACIÓN EN EL CICLO DE VIDA |
| 03 | INFORMACIÓN Y HERRAMIENTAS | 10 | FORMA DE EVALUAR |
| 04 | SEGUIMIENTO DE ERRORES CON JIRA | 11 | CÓMO SABEMOS QUE FUNCIONA |
| 05 | DIFERENCIA CON EL ANÁLISIS DE
CÓDIGO ESTÁTICO | 12 | INTELIGENCIA ARTIFICIAL EN ESTE
CONTEXTO |
| 06 | OPTIMIZAR LA ORGANIZACIÓN Y LA
ARQUITECTURA | 13 | CÓMO EMPEZAR |
| 07 | CÓDIGO HEREDADO | 14 | CODEMAAT |

01. ¿QUÉ ES?



La idea del código desde el lado de las personas.



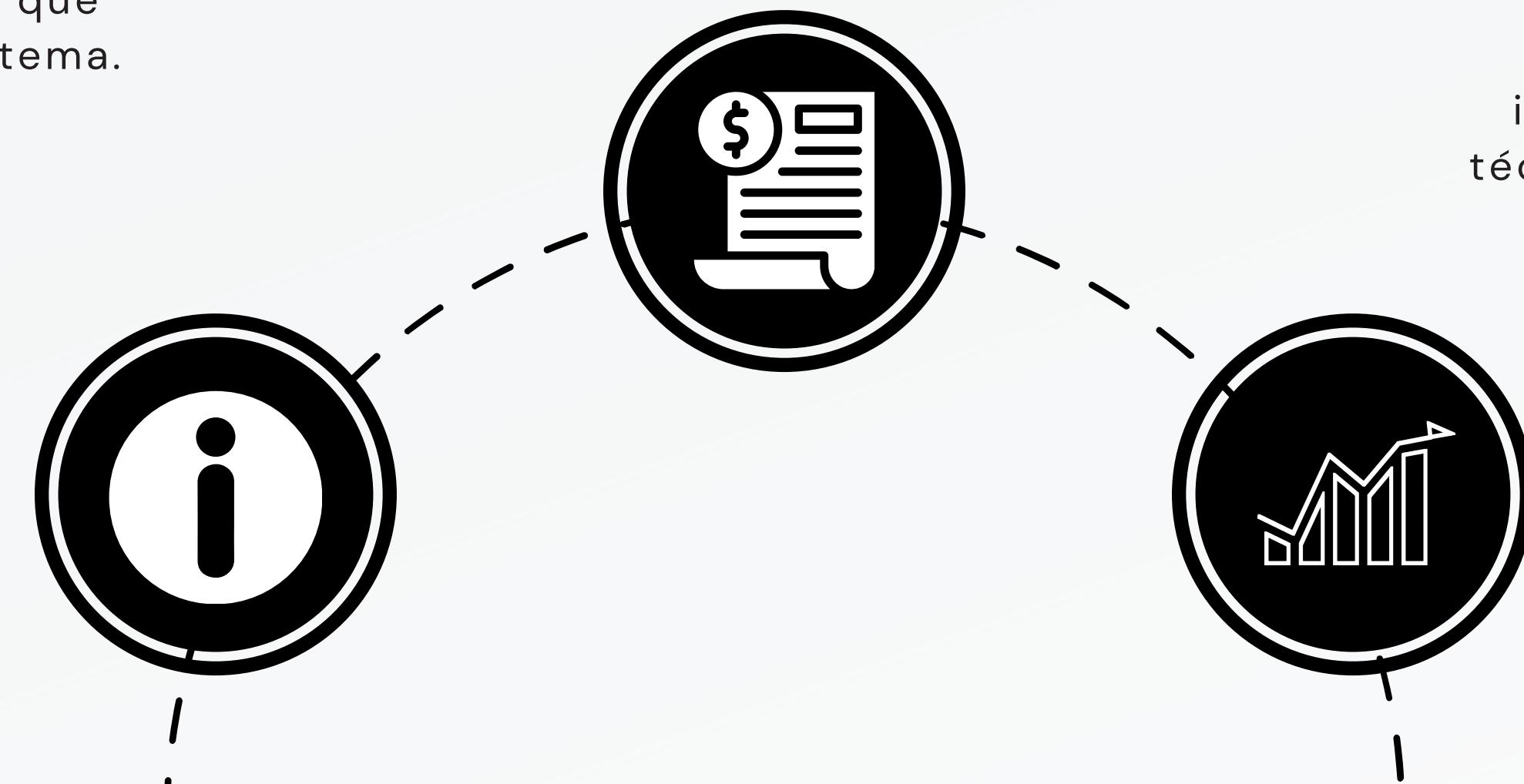
Como la organización y los desarrolladores han desarrollado el código.



02. ¿QUE PROBLEMAS AYUDA A RESOLVER?

Información adicional

Podremos saber el comportamiento de la dinámica de lo que ocurre en el sistema.



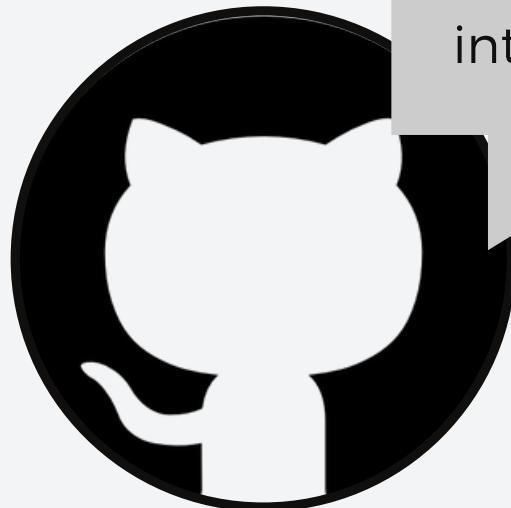
Deudas técnicas

Identifica y prioriza las deudas técnicas que contenga el código.

Análisis de puntos críticos

Observar el patrón de comportamiento de las partes con las que trabajamos, para identificar la deuda técnica que represente un mayor riesgo

03. INFORMACIÓN Y HERRAMIENTAS



GITHUB



Podemos obtener toda la información de como los desarrolladores han interactuado con el código.



JIRA



Herramientas de gestión de productos.

04. SEGUIMIENTO DE ERRORES CON Jira

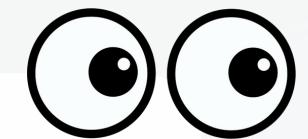
Principal problema: a menudo es necesario presentar argumentos comerciales a los líderes técnicos y a los gerentes.



Estas personas no
están familiarizadas
con el código



También podremos
visualizar el retorno
de la inversión si
pagamos esa deuda
técnica.



Utilizar
herramientas que
puedan mostrarnos
eso de forma más
sencilla.

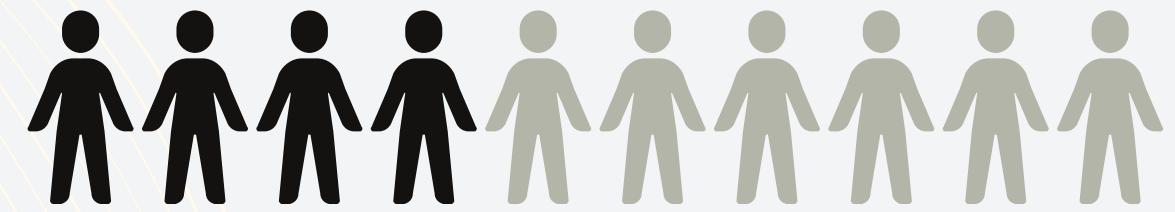


Unimos la parte
técnica con el
negocio.

ESTADÍSTICAS

Tiempo de los programadores peleándose con deudas técnicas y código malo en las empresas.

23-42%



05. DIFERENCIA CON EL ANÁLISIS DE CÓDIGO ESTÁTICO

CÓDIGO ESTÁTICO



- Identifica mal código, no te dice el impacto que tiene.
- Se limita al código en sí.

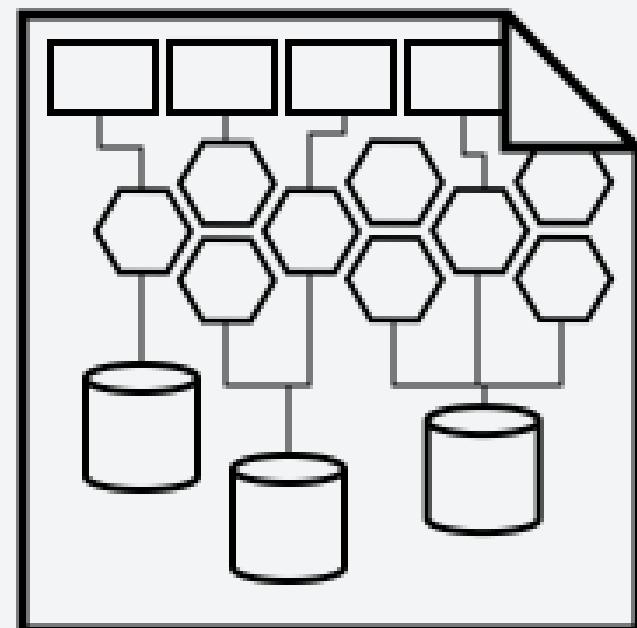
- Nos informa del impacto que tiene el código y si es una deuda técnica.
- También analiza la organización que construye el código

COMPORTAMIENTO



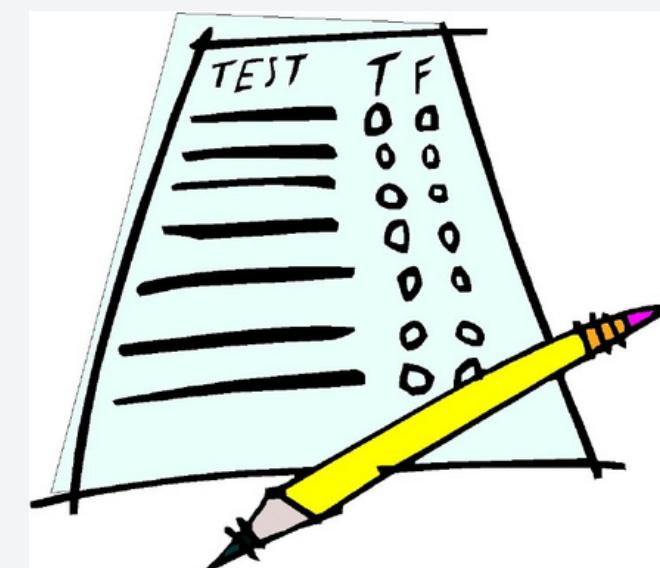
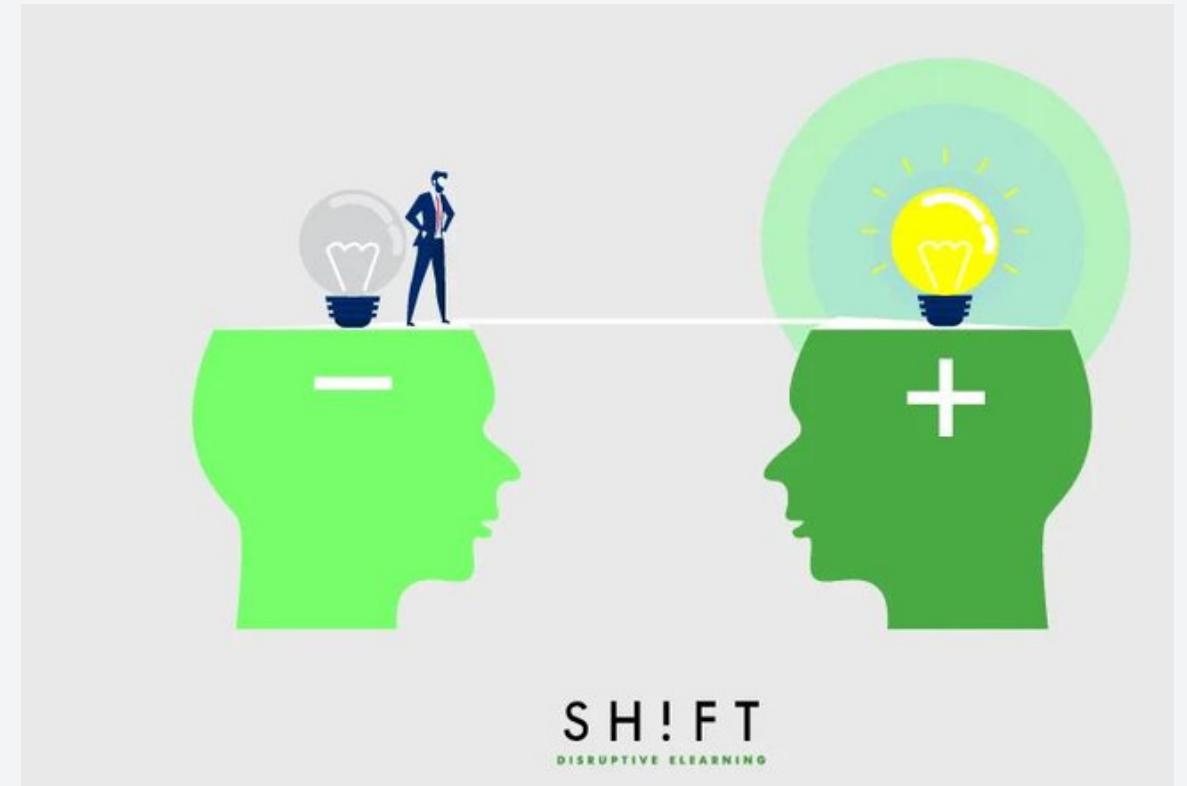
06. OPTIMIZAR LA ORGANIZACIÓN Y LA ARQUITECTURA

- ALINEACIÓN ARQUITECTURA-EQUIPO
 - COMPONENTES
 - FEATURE TEAMS
- DISCREPANCIAS ENTRE EQUIPO Y ARQUITECTURA CAUSAN:
 - CUELLOS DE BOTELLA
 - PROBLEMAS DE FUSIÓN DE RAMAS
- CAMBIAR ORGANIZACIÓN DEL EQUIPO -> EVOLUCIONAR ARQUITECTURA



07. CÓDIGO HEREDADO

- ANALIZAR BRECHAS DE CONOCIMIENTO
- NO FAMILIARIDAD -> MÁS DIFÍCIL AUN CON MISMA COMPLEJIDAD
- CÓDIGO HEREDADO PUEDE SER MALO A PESAR DE SUS PRUEBAS
- PRUEBAS MAL HECHAS = NO PRUEBAS
 - CAUSAS:
 - FALTA EXPERIENCIA
 - NECESIDAD DE LLEGAR A COBERTURA DE CÓDIGO MÍNIMA



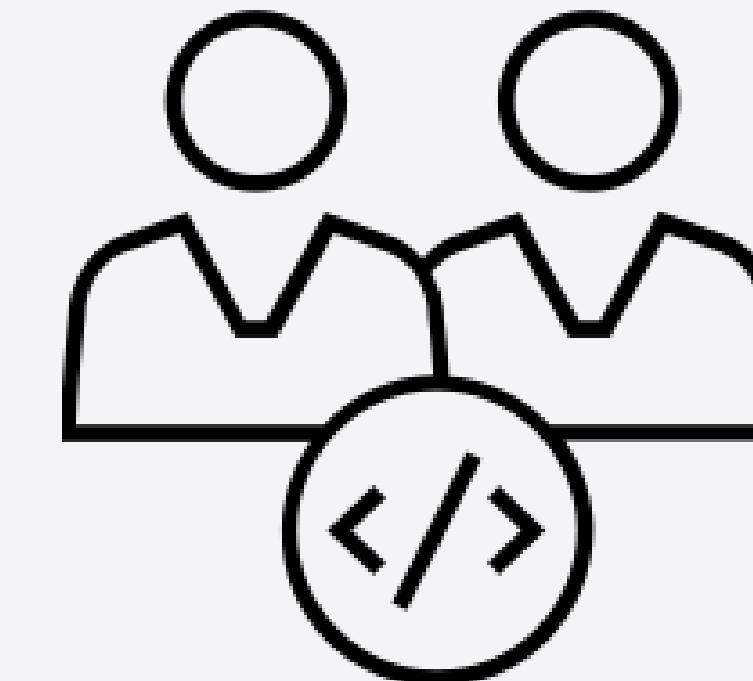
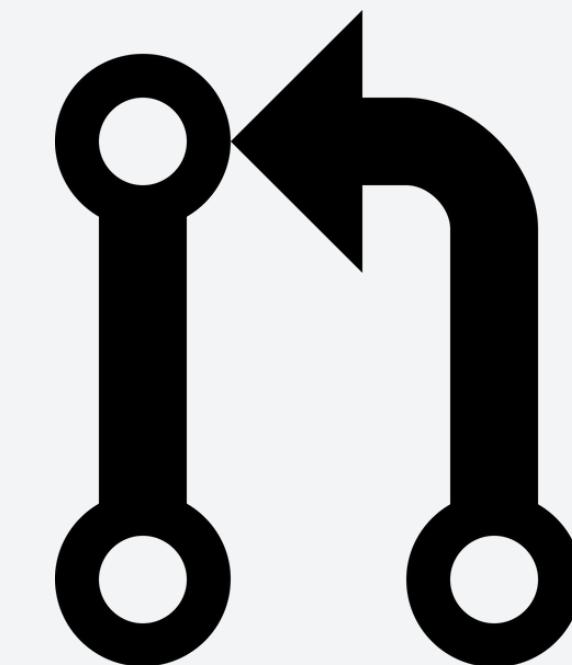
08. PROYECTOS GREENFIELD

- DATOS DISPONIBLES A LAS pocas SEMANAS
- DETECTA A TIEMPO PROBLEMAS
- AYUDA VER BAJADAS DE CALIDAD
 - EJEMPLO: POCO TIEMPO PARA UNA FECHA DE ENTREGA



09. INTEGRACIÓN EN EL CICLO DE VIDA

- CADA PULLREQUEST: IMPIDE QUE CREZCA LA DEUDA TÉCNICA
- MEDIO DE COMUNICACIÓN EN REUNIONES, RETROSPECTIVAS.
- PAIR PROGRAMMING: AÑADIR ETIQUETAS DE COAUTORES PARA FACILITAR EL ANÁLISIS



10. FORMA DE EVALUAR



Ético

No es ético
evaluar
individualmente



Equipo

Los
desarrolladores
no trabajarían en
equipo



Objetivo

No iremos hacia
un único
objetivo

11.1. CÓMO SABEMOS QUE FUNCIONA



IDENTIFICACIÓN
DE PATRONES



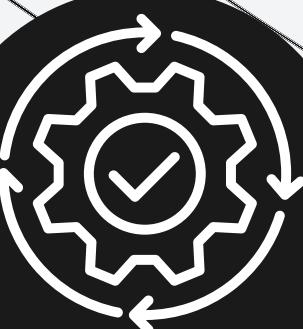
DETENCIÓN DE
ANOMALIAS



MEJORA DE LA
CALIDAD DEL
CÓDIGO



SEGURIDAD



OPTIMIZACIÓN
DEL
RENDIMIENTO



AUTOMATIZACIÓN

11.2. CÓMO LO USAMOS



CÓDIGO NUEVO



CÓDIGO ANTIGUO (BASE)



ANÁLISIS DE PUNTOS DE ACCESO



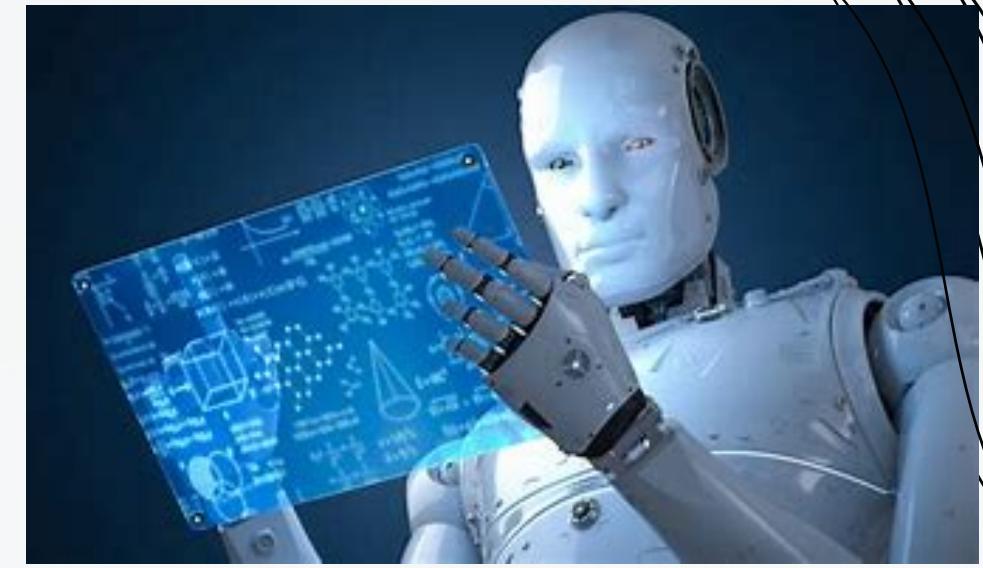
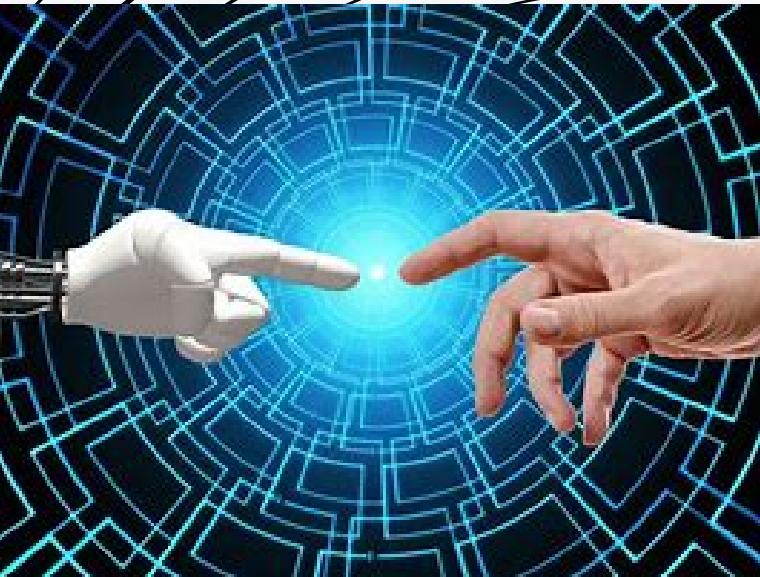
12 INTELIGENCIA ARTIFICIAL

PRIORIZACION DE
PULL REQUESTS

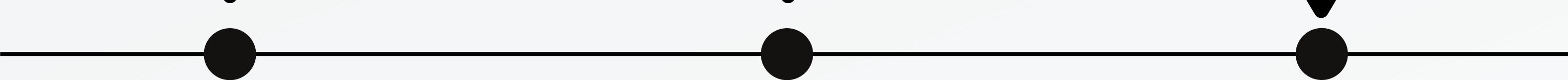
ASISTENCIA EN
LA REVISION DEL
CÓDIGO

PREDICCIÓN DEL
RIESGO DE
DEFECTOS

CONSIDERACIÓN
DE LA
DIMENSIÓN
SOCIAL



13. ¿POR DONDE EMPIEZO?



PUNTOS CRÍTICOS

Parte más simple
La base para cualquier
otro análisis

ACOPLAMIENTO DE CAMBIOS

Los componentes son
simples, ¿pero lo es el
sistema?

CUESTIONES ORGANIZATIVAS

¿Qué pasa si una
persona deja el
proyecto?
Distribución del
conocimiento

14. CODE MAAT

- Herramienta Open Source.
- Compatible con cualquier control de versiones.
- Otras alternativas: CodeScene y el propio Git.

Options:

-l, --log LOG	authors
-c, --version-control VCS	
-a, --analysis ANALYSIS	
--input-encoding INPUT-ENCODING	
-r, --rows ROWS	
-g, --group GROUP	
-n, --min-revs MIN-REVS	5
-m, --min-shared-revs MIN-SHARED-REVS	5
-i, --min-coupling MIN-COUPLING	30
-x, --max-coupling MAX-COUPLING	100
-s, --max-changeset-size MAX-CHANGESET-SIZE	30
-e, --expression-to-match MATCH-EXPRESSION	
-t, --temporal-period TEMPORAL-PERIOD	
-d, --age-time-now AGE-TIME_NOW	
-h, --help	

**GRACIAS
A TODOS**

¿PREGUNTAS?

