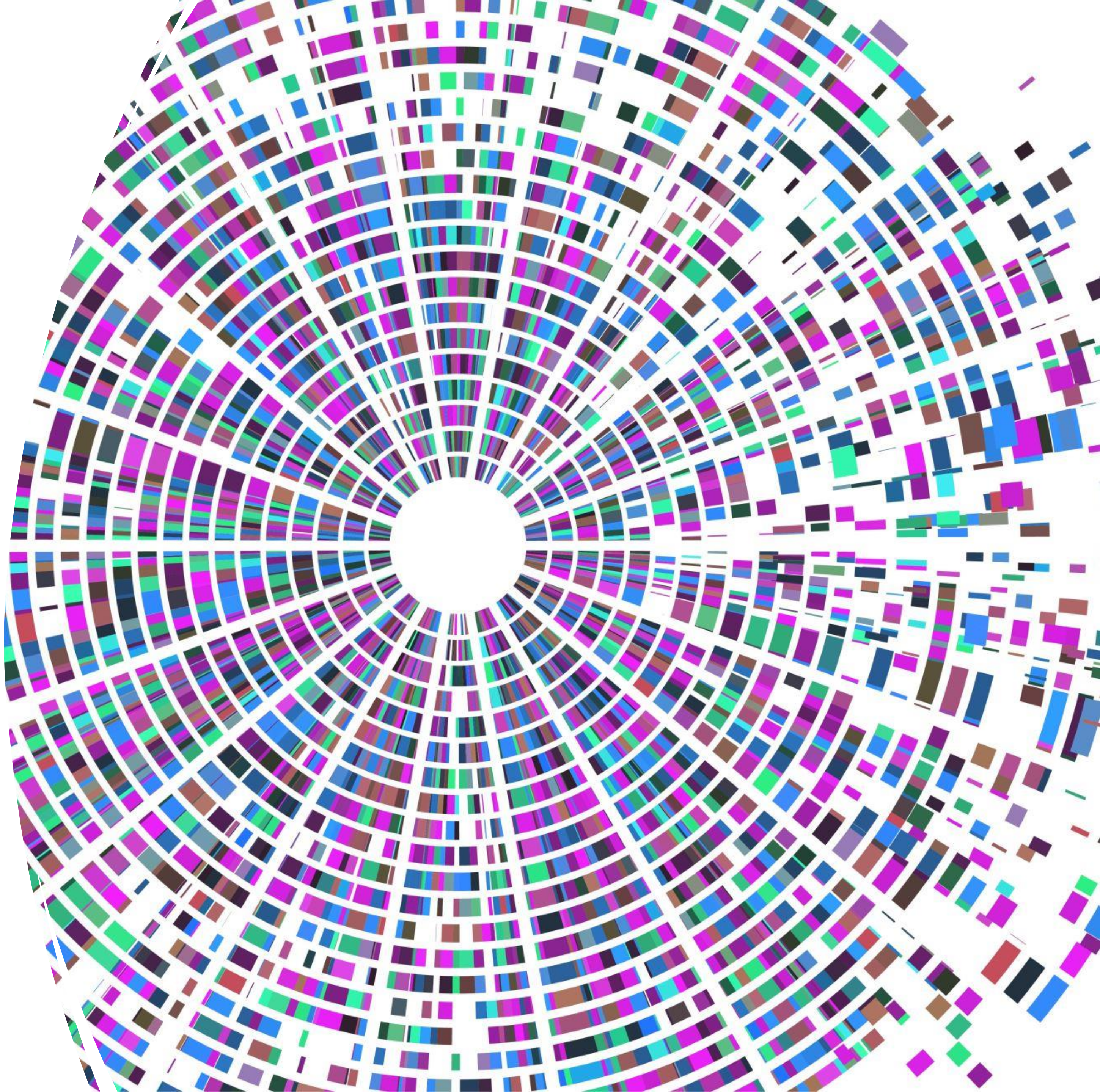
Behaviour Code Analisis



Realizado por:

Adrián Fernández Alonso UO264268

Javier López de Juan UO271447

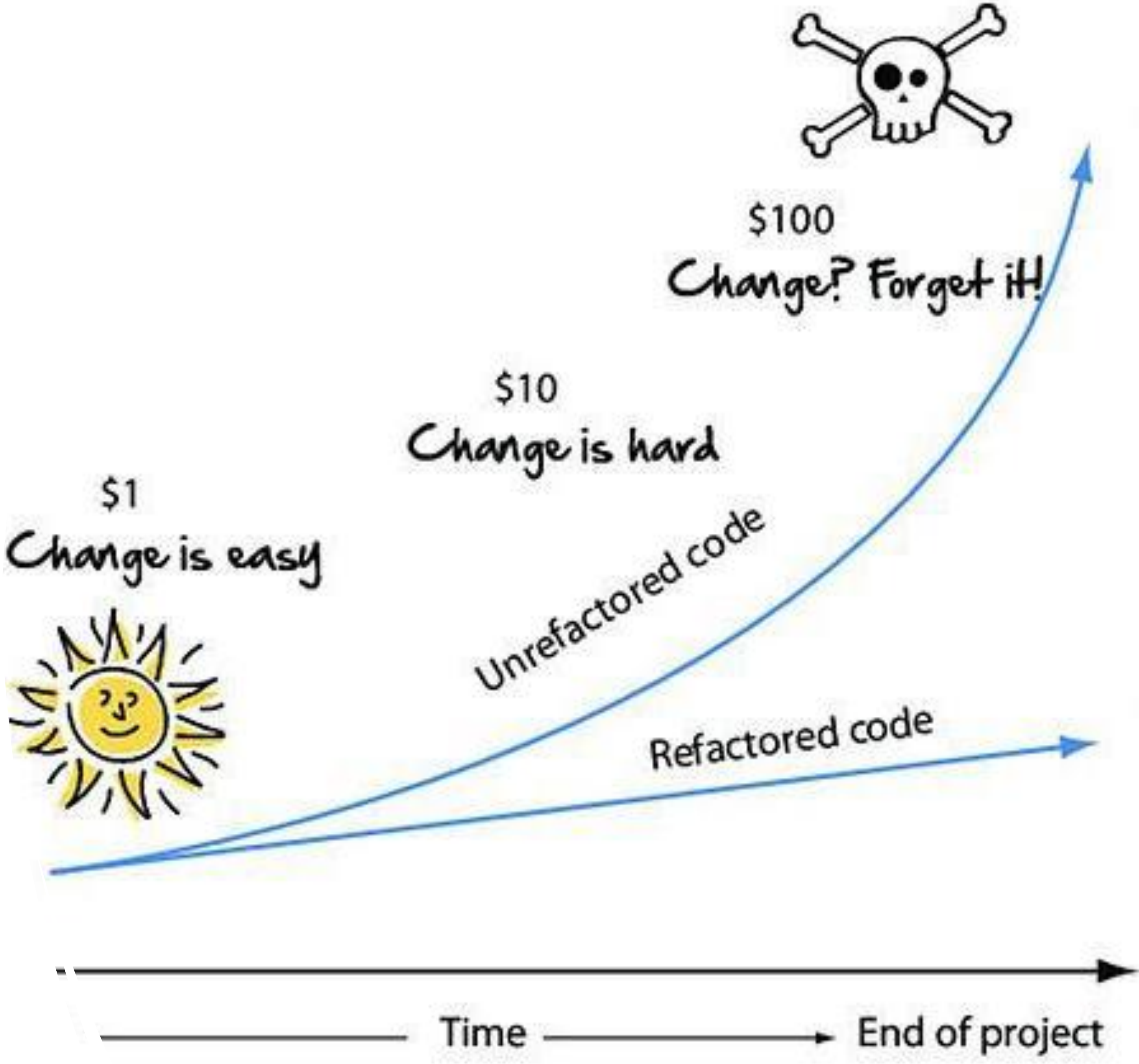
¿Qué es Behaviour Code Analisis?

- Identifica cómo se interactúa con el código.
- Permite priorizar la deuda técnica.
- No analizar los archivos individualmente, si no en conjunto.

Motivación

- ¿Cómo podemos mantener grandes sistemas con cierta complejidad?
- ¿Qué partes necesitan refactorización?
- ¿Qué partes es probable que contengan errores?
- ¿Cuáles son las deudas técnicas y cómo comunicarlas a las partes interesadas?
- ¿Qué métricas usamos para apoyarnos?

La deuda técnica



El problema no es técnico, es social

- Efecto espectador: El grupo nos influye.
  - Ignorancia pluralista
  - Difusión de la responsabilidad

Técnicas de análisis del comportamiento del código

- Hay 3:
  - Hotspots.
  - Tendencia a la complejidad.
  - Cambio de acoplamiento (**Change Coupling)**

HOTSPOTS

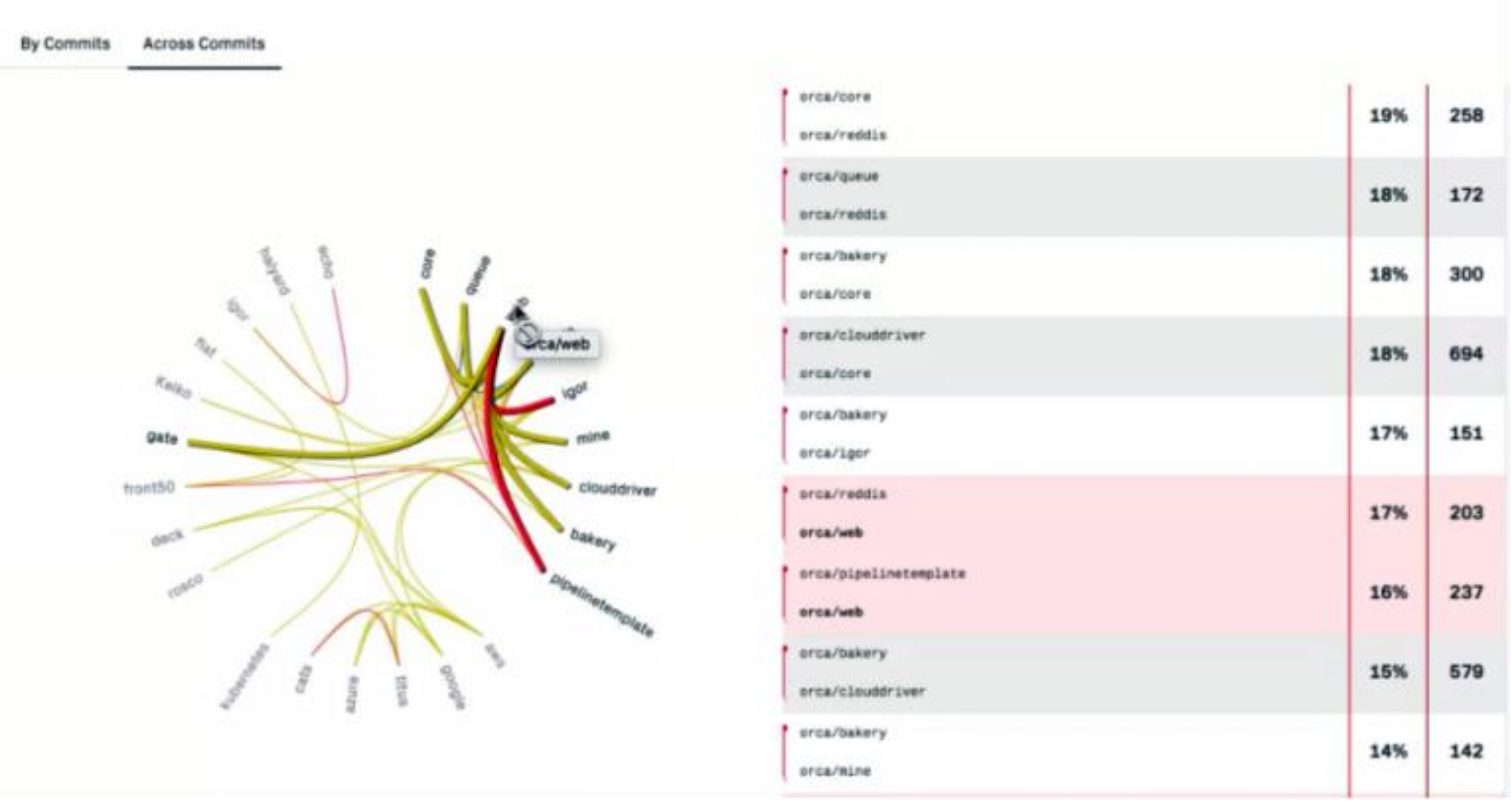
OBJETIVOS DE REFACTORIZACIÓN

PUEDEN CONTENER MUCHOS ERRORES

RESULTADOS DE CODIGO ACTIVO Y COMPLEJO.

Tendencia a la complejidad

Cambio de acoplamiento



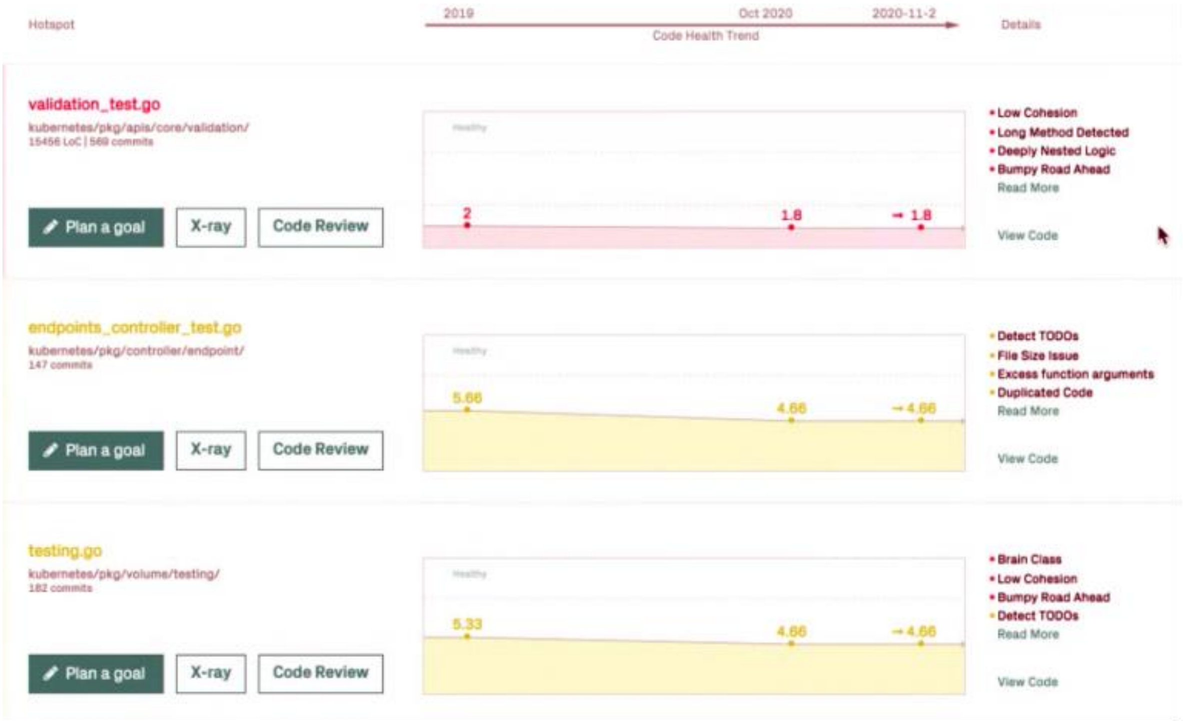
orca/core orca/reddis	19%	258
orca/queue orca/reddis	18%	172
orca/bakery orca/core	18%	300
orca/clouddriver orca/core	18%	694
orca/bakery orca/igor	17%	151
orca/reddis **orca/web**	17%	203
orca/pipelinetemplate **orca/web**	16%	237
orca/bakery orca/clouddriver	15%	579
orca/bakery orca/mine	14%	142

Métricas

HEALTHINESS (SALUBRIDAD)

COMPLEJIDAD

Healthiness



validation_test.go
kubernetes/pkg/apis/core/validation/
15456 LoC | 569 commits

- Low Cohesion
- Long Method Detected
- Deeply Nested Logic
- Bumpy Road Ahead

Read More

View Code

endpoints_controller_test.go
kubernetes/pkg/controller/endpoint/
147 commits

- Detect TODOs
- File Size Issue
- Excess function arguments
- Duplicated Code

Read More

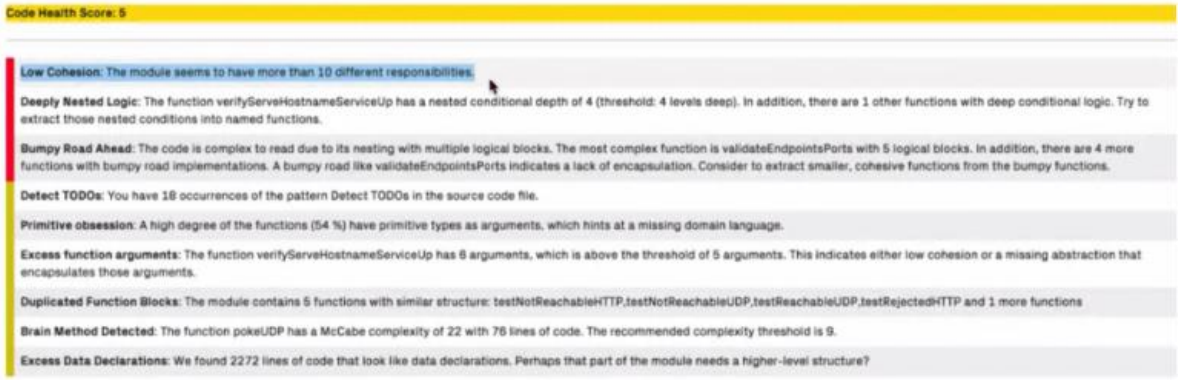
View Code

testing.go
kubernetes/pkg/volume/testing/
182 commits

- Brain Class
- Low Cohesion
- Bumpy Road Ahead
- Detect TODOs

Read More

View Code



Code Health Score: 5

Low Cohesion: The module seems to have more than 10 different responsibilities.

Deeply Nested Logic: The function verifyServeHostnameServiceUp has a nested conditional depth of 4 (threshold: 4 levels deep). In addition, there are 1 other functions with deep conditional logic. Try to extract those nested conditions into named functions.

Bumpy Road Ahead: The code is complex to read due to its nesting with multiple logical blocks. The most complex function is validateEndpointsPorts with 5 logical blocks. In addition, there are 4 more functions with bumpy road implementations. A bumpy road like validateEndpointsPorts indicates a lack of encapsulation. Consider to extract smaller, cohesive functions from the bumpy functions.

Detect TODOs: You have 18 occurrences of the pattern Detect TODOs in the source code file.

Primitive obsession: A high degree of the functions (54 %) have primitive types as arguments, which hints at a missing domain language.

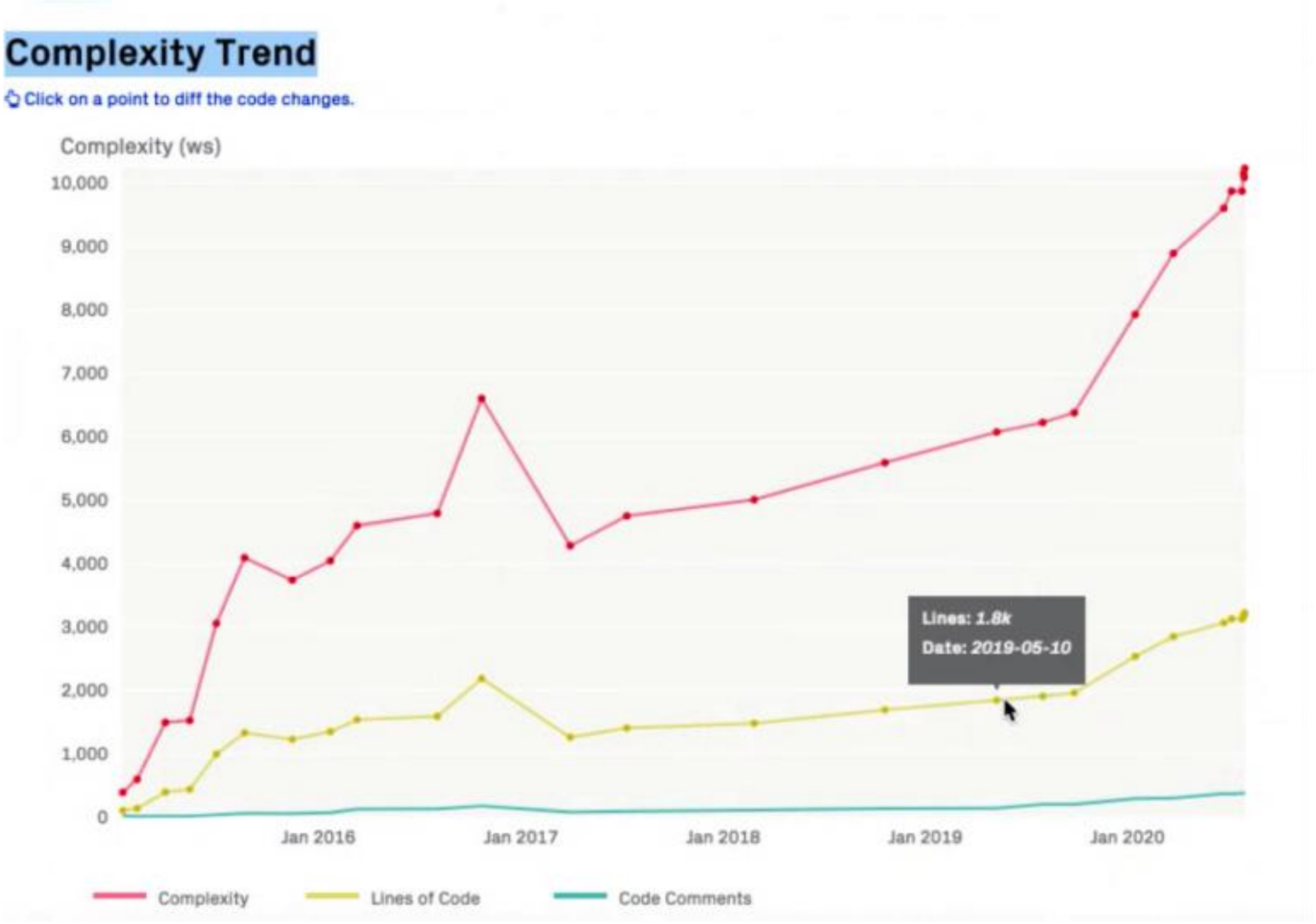
Excess function arguments: The function verifyServeHostnameServiceUp has 6 arguments, which is above the threshold of 5 arguments. This indicates either low cohesion or a missing abstraction that encapsulates those arguments.

Duplicated Function Blocks: The module contains 5 functions with similar structure: testNotReachableHTTP,testNotReachableUDP,testReachableUDP,testRejectedHTTP and 1 more functions

Brain Method Detected: The function pokeUDP has a McCabe complexity of 22 with 76 lines of code. The recommended complexity threshold is 9.

Excess Data Declarations: We found 2272 lines of code that look like data declarations. Perhaps that part of the module needs a higher-level structure?

Complejidad



Complexity Trend

Click on a point to diff the code changes.

¿PREGUNTAS?