



Curso de

# Flujo Moderno de Desarrollo de Software

Claudio Pinkus

*Co-fundador, CodeStream*  
[codestream.com](http://codestream.com)



# Contenido

- **Clase 1:** El flujo de trabajo y el principio *Shift Left* (basado en Agile).
- **Clase 2:** El editor de texto es tu centro de control.
- **Clase 3:** El flujo moderno.
- **Clase 4:** GitHub en tu editor (VS Code).



# Contenido

- **Clase 5:** Gestión de tareas (Jira).
- **Clase 6:** Colaboración y comunicación (Slack).
- **Clase 7:** Trabajo remoto y transparencia.
- **Clase 8:** La documentación y el flujo moderno.
- **Clase 9:** El futuro del desarrollo de software.



# Contenido (Práctica)

- **Práctica I:** Instalación de CodeStream y GitHub en VS Code.
- **Práctica II:** Pull Request integrado en VS Code.
- **Práctica III:** Feedback Request, Jira.
- **Práctica IV:** Code Chat, Slack.
- **Práctica V:** Documentación, transparencia.
- **Práctica VI:** Project Open Source.



# Para quién es este curso

- Desarrolladores que ya saben programar y han completado algún proyecto, sin tener que ser un desarrollador profesional.
- Desarrolladores que quieren actualizar sus conocimientos.
- Cualquier desarrollador que quiera mejorar la calidad del código con menos esfuerzo.
- Líderes de equipos de desarrolladores.

# Herramientas necesarias

- Un editor de texto moderno (VS Code\*, Visual Studio o uno de JetBrains).
- Un sistema Git (GitHub\*, GitLab, BitBucket).
- Un sistema de gestión de proyectos (JIRA\*, Trello, GitHub Issues, etc.).
- Un sistema de Mensajería de equipos (Slack\*, MS Teams) o correo electrónico.

---

# El flujo de trabajo y el principio *Shift Left*





# Definición: flow (flujo)

## What is flow?

According to the book PeopleWare by Tom DeMarco and Timothy Lister, they define “flow” on page 63 as:

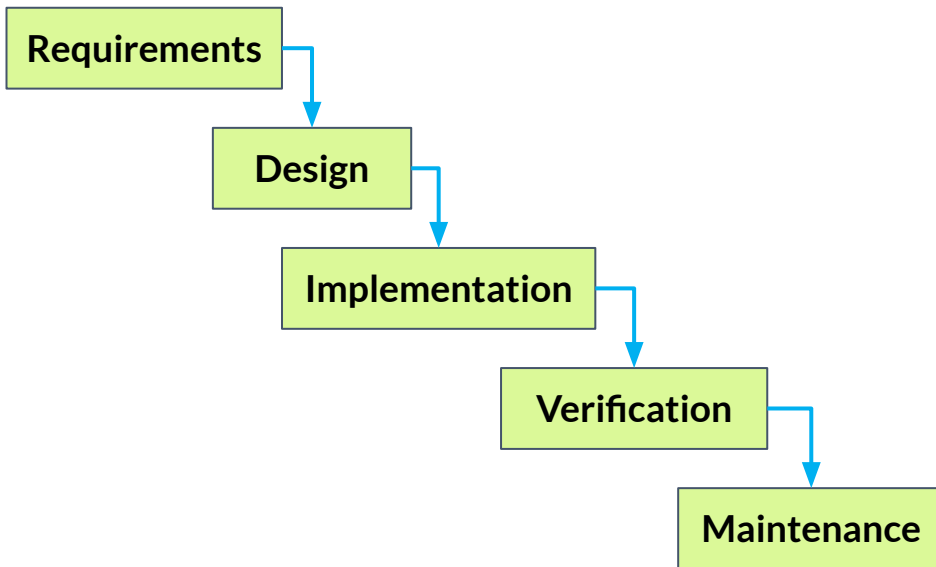
“A condition of deep, nearly meditative involvement. In this state there is a gentle sense of euphoria, and one is largely unaware of the passage of time: ‘I began to work, I looked up and three hours had passed.’ There is no consciousness of effort; the work seems to well, flow.

# Definición: workflow (flujo de trabajo)

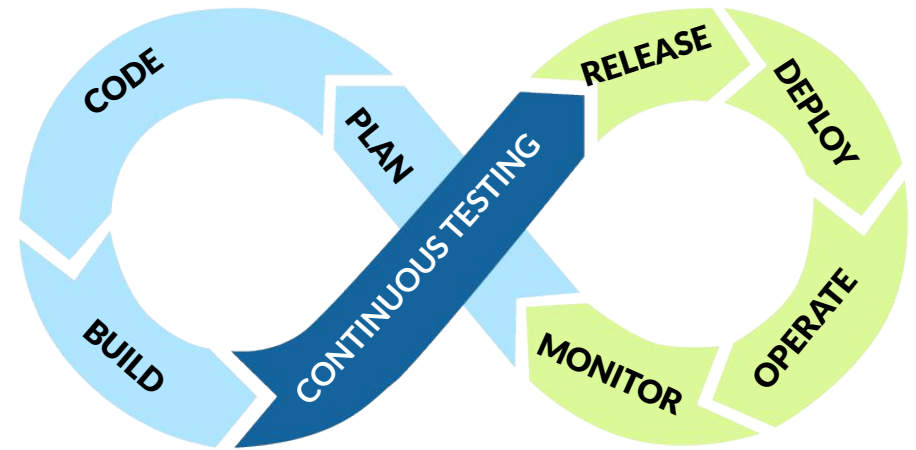
## Definition of Workflow

A **Workflow** is a sequence of tasks that processes a set of data. Workflows occur across every kind of business and industry. Anytime data is passed between humans and/or systems, a workflow is created. Workflows are the paths that describe how something goes from being undone to done, or raw to processed.

# Ciclo de vida en software

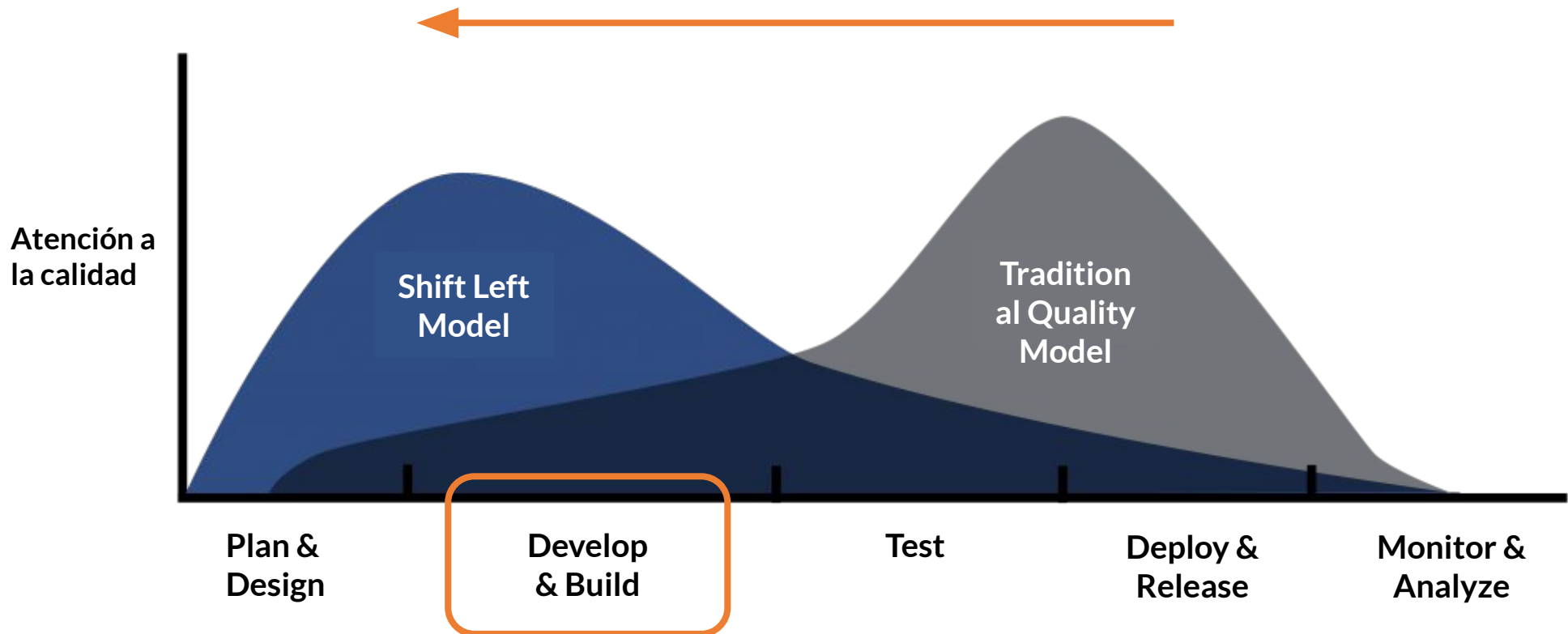


Cascada



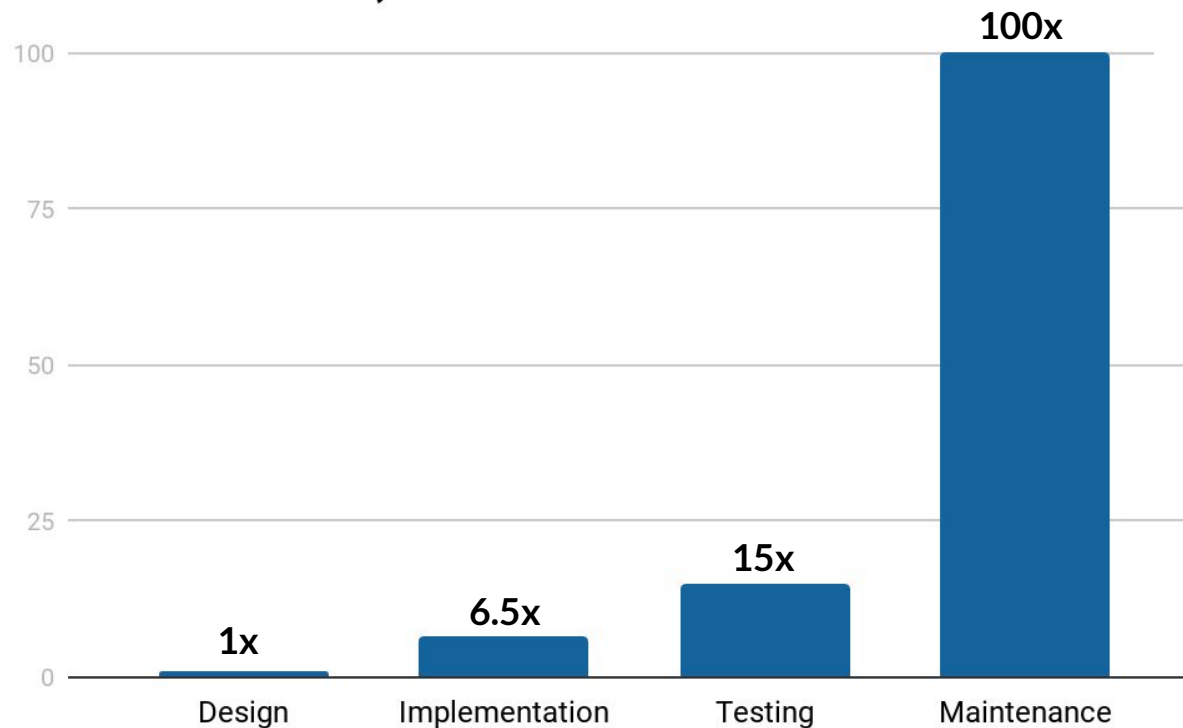
Moderno

# El principio Shift Left (desplazarse a la izquierda)



# Beneficios del flujo moderno

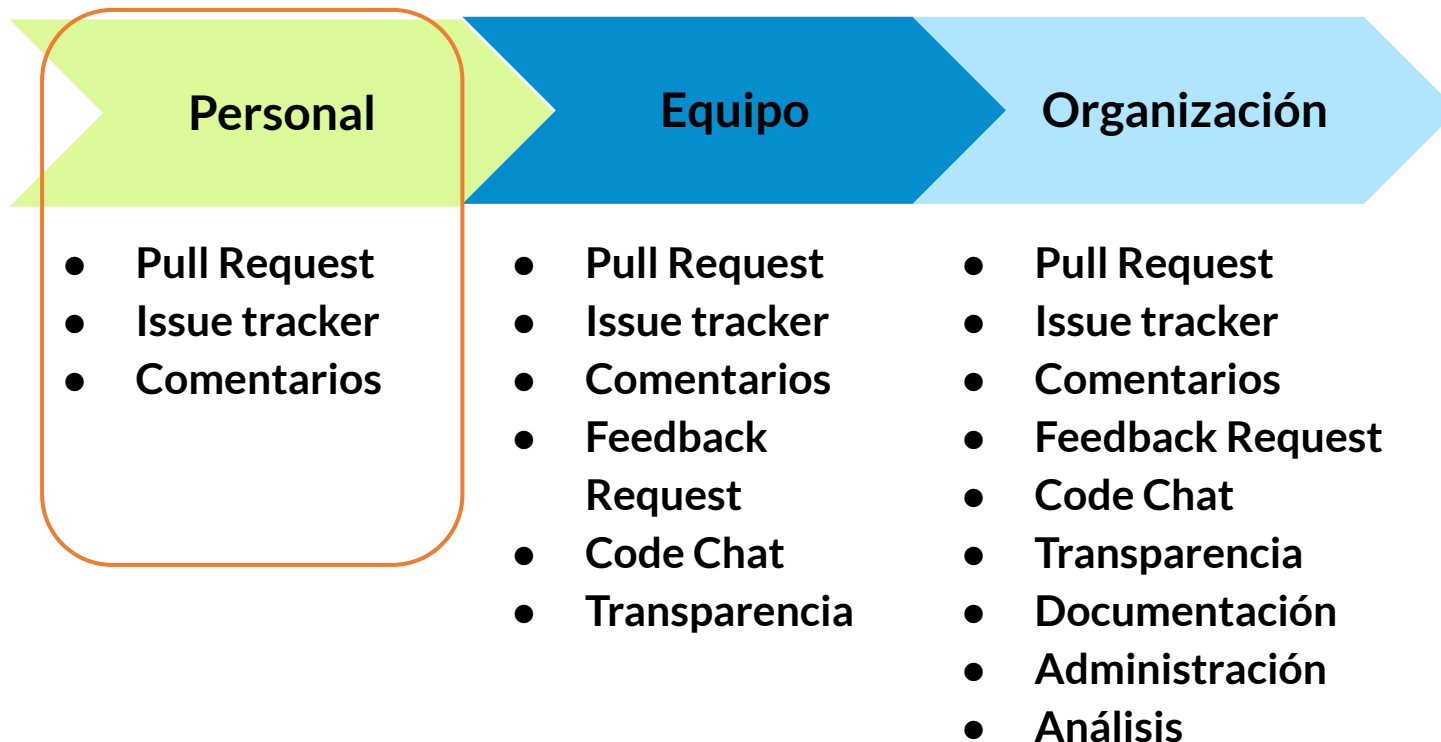
**Relative Costs to Fix Software Defects (Source: IBM Systems Sciences Institute)**



**Phase/Stage of the S/W Development in Which the Defect is found**

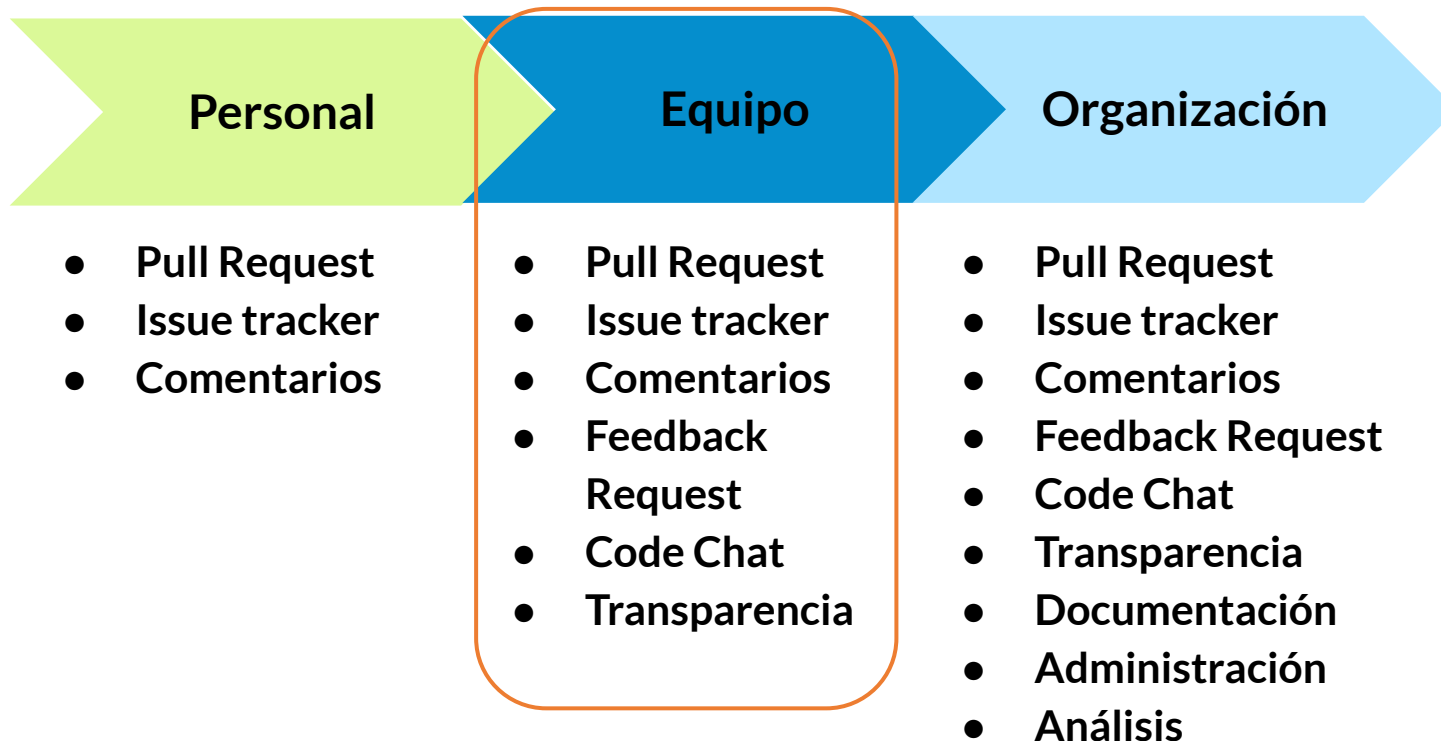
# Implementación del flujo moderno

## Uso/Efecto



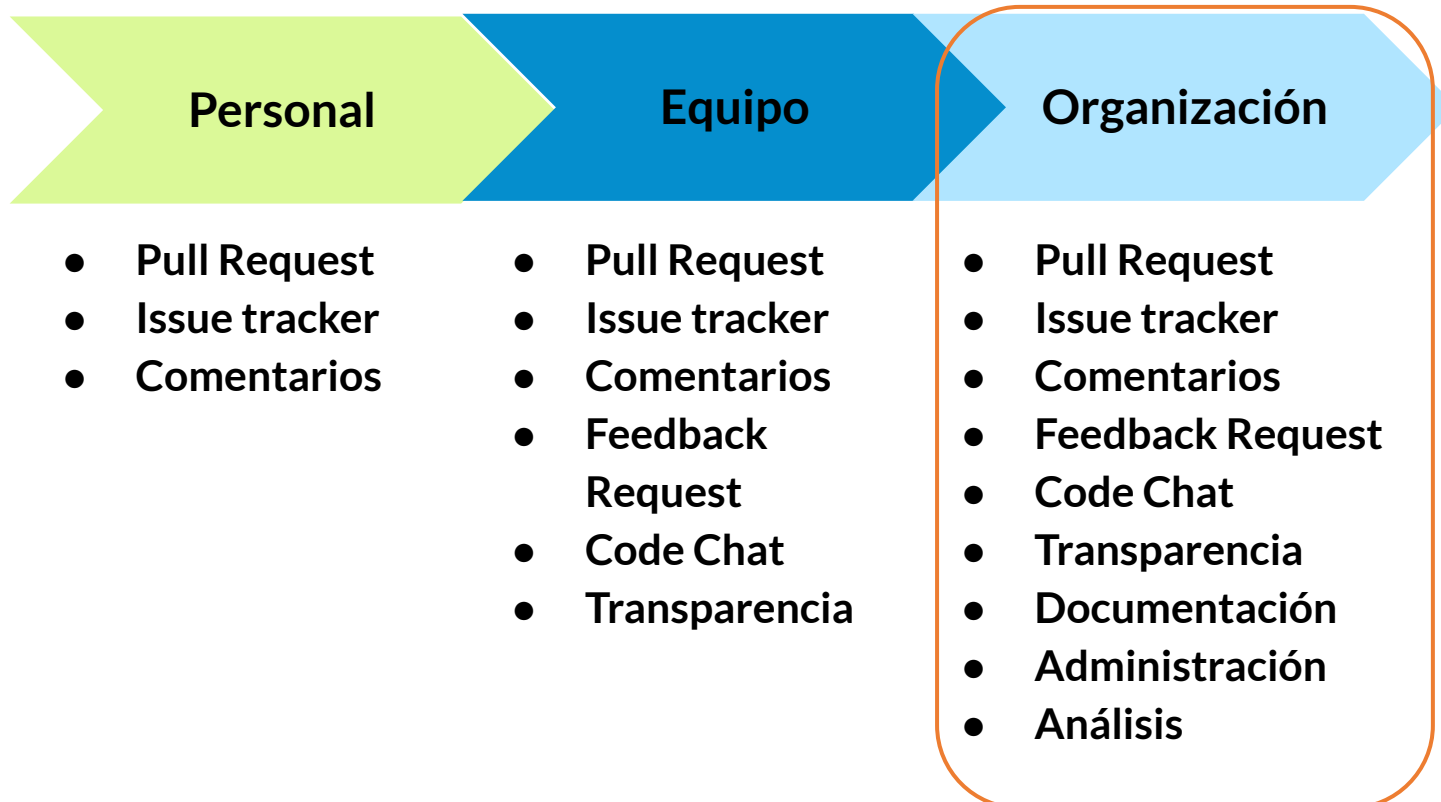
# Implementación del flujo moderno

## Uso/Efecto



# Implementación del flujo moderno

## Uso/Efecto







# Frecuencia de colaboración

- Tradicional: se hace una revisión de código *al terminar* el desarrollo.
- Moderno: se hacen muchas revisiones de código pequeñas *a medida* que se va desarrollando el código.

# Prácticas de Ingeniería de Google

- Documentado y disponible en <https://google.github.io/eng-practices/>.
- Revisiones de código en menos de 24 horas (Promedio Google: en menos de 4 horas).
- Revisiones con menos código: Change Lists (CL) digeribles rápidamente, menos de 100 líneas (Promedio Google: 24 líneas).

# Cómo aplicar *Shift Left* en desarrollo

- **Colaboración hiper-frecuente**  
Más preguntas antes, más comentarios antes.
- **Integración de las herramientas para evitar cambio de contexto**  
No saltar de una herramienta a otra.
- **La calidad se incorpora en el proceso**  
Tener consenso y aplicar revisiones lo antes posible.

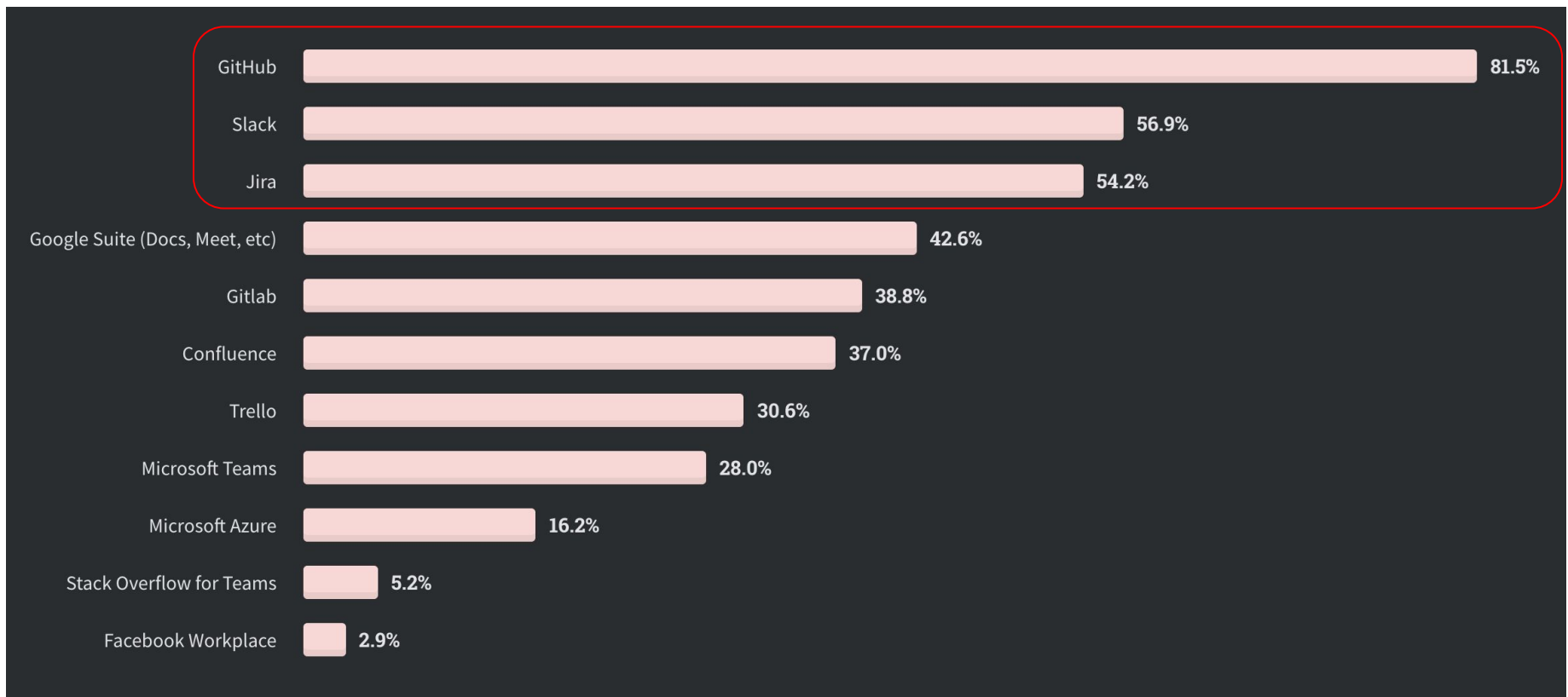
---

**El editor de texto es  
tu centro de control**

# El editor del texto como Eje (Hub)

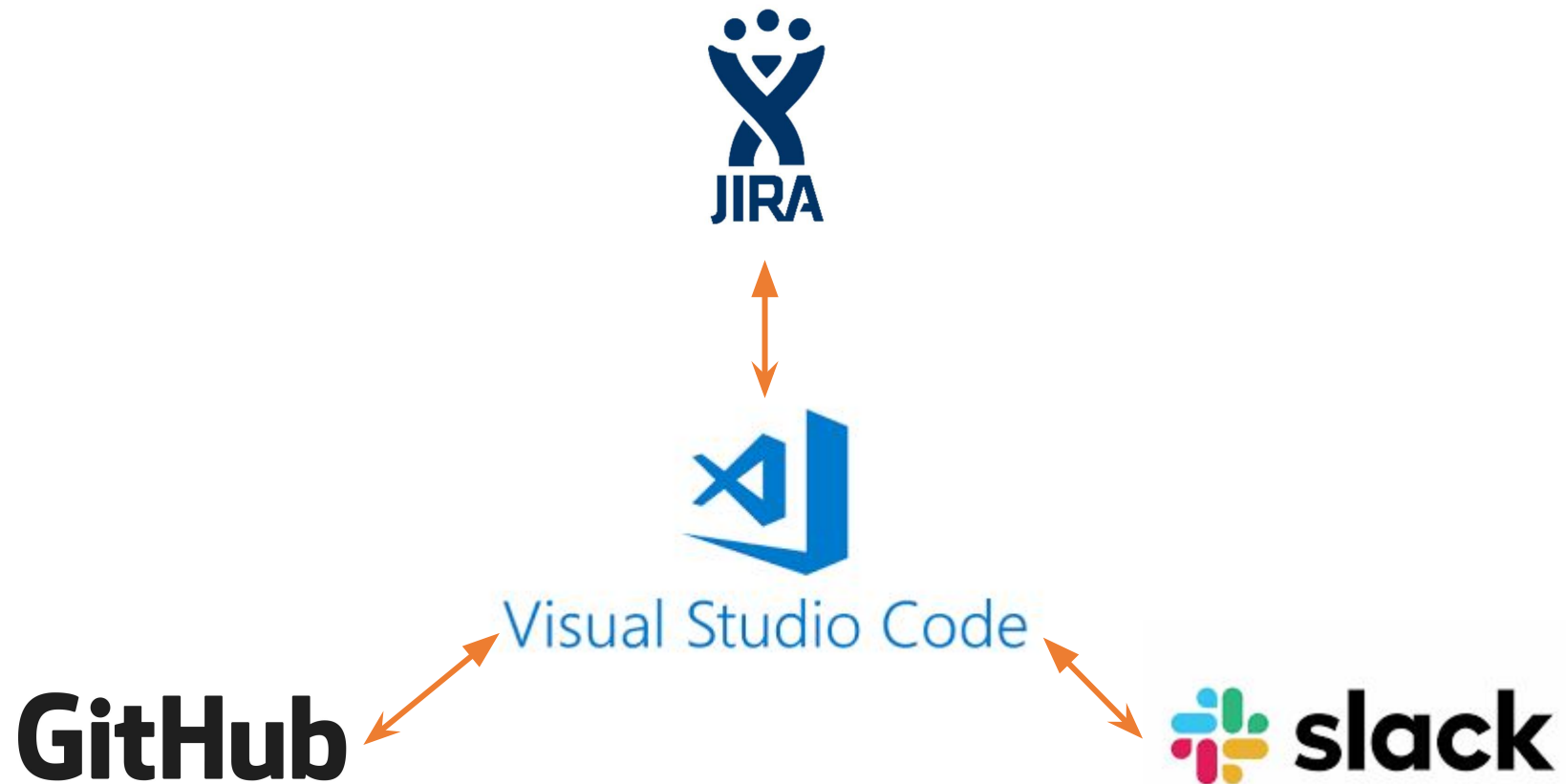
- Tu trabajas todo el día en tu editor de texto:
  - VS Code, IntelliJ IDEA, PyCharm...
- Y usas otras herramientas que no están integradas a tu editor:
  - GitHub - Jira - Slack - email
- Cambios de contexto reducen la productividad.

# Encuesta: ¿Qué herramientas utilizas?\*



\*Stack Overflow Survey 2020

# El editor del texto como Eje





# Integración de herramientas

- Todos los editores de texto modernos son extensibles:
  - VS Code
  - IntelliJ IDEA (y todos los editores de JetBrains)
  - PyCharm
  - Visual Studio
  - Android Studio
- La idea esencial es mejorar la productividad eliminando pasos innecesarios.



# Ejemplo 1: Flujo integrado

Sin integración

20

Pasos

Con integración

6

Pasos

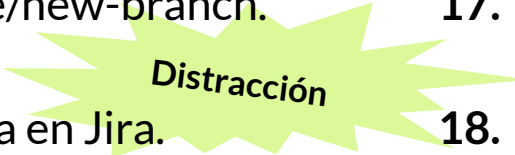


# Ejemplo 1: Flujo integrado

## Sin integración

1. Escribir código.
2. Alt-tab a terminal.
3. Git status.
4. Verificar los cambios archivo por archivo en el editor.
5. Git diff.
6. Git branch -b feature/new-branch.
7. Alt-tab a Chrome.
8. Encontrar la lengüeta en Jira.
9. Ir a la lista de issues.
10. Encontrar el issue en cuestión.
11. Copiar.  
Alt-tab, back to terminal.
12. Git commit -am 'commit message goes here #JIRA-TICKET'.
13. Git pull.
14. Git push.
15. Copiar el URL de la terminal.
16. Alt-tab a Chrome. Pegar.
17. Crear un título para el PR y una descripción, RETURN.
18. Agregar un revisor al PR.
19. Alt-tab a Slack.
20. @mention al revisor para avisarle qué estás esperando.

# Ejemplo 1: Flujo integrado

## Sin integración

1. Escribir código.
2. Alt-tab a terminal.
3. Git status.
4. Verificar los cambios archivo por archivo en el editor.
5. Git diff.
6. Git branch -b feature/new-branch.
7. Alt-tab a Chrome. 
8. Encontrar la lengüeta en Jira.
9. Ir a la lista de issues.
10. Encontrar el issue en cuestión.
11. Copiar.  
Alt-tab, back to terminal.
12. Git commit -am 'commit message goes here #JIRA-TICKET'.
13. Git pull.
14. Git push.
15. Copiar el URL de la terminal. 
16. Alt-tab a Chrome. Pegar.
17. Crear un título para el PR y una descripción, RETURN.
18. Agregar un revisor al PR. 
19. Alt-tab a Slack.
20. @mention al revisor para avisarle qué estás esperando.

# Ejemplo 1: Flujo integrado

## Con integración

1. Hacer clic en el ticket para generar una rama y empezar a trabajar.
2. Escribir código.
3. Realizar revisión, agregar y commit a cada archivo en el panel SCM del editor.
4. Sincronizar los cambios a GitHub.
5. Hacer clic en New Pull Request (el título y descripción se crean automáticamente con referencia al ticket de Jira).
6. Agregar el revisor al PR.

## Ejemplo 2: GitHub (Revisor)

Sin integración

14

Pasos

Con integración

4

Pasos

# Ejemplo 2: GitHub (revisor)

## Sin integración

1. Recibir notificación.
2. Clic para cargar PR en github.com.
3. Examinar el review, darte cuenta que tienes que mirar el código en contexto para entender los cambios.
4. Copiar el nombre de la rama.
5. Alt-tab a terminal, encontrar el directorio donde está el repositorio.
6. Git fetch.
7. Git checkout branch-name.
8. Git pull.
9. Alt-tab a tu editor, donde puedes ver el código.
10. Alt-tab de vuelta a github.com, navegar a cualquier archivo en el que quieras hacer un comentario, y agregar el comentario.
11. Seguir alternando ida y vuelta entre el editor y github.com.
12. Terminar la revisión.
13. Alt-tab a Slack.
14. @mention al autor para avisarle que has terminado la revisión.

# Ejemplo 2: GitHub (revisor)

## Con integración

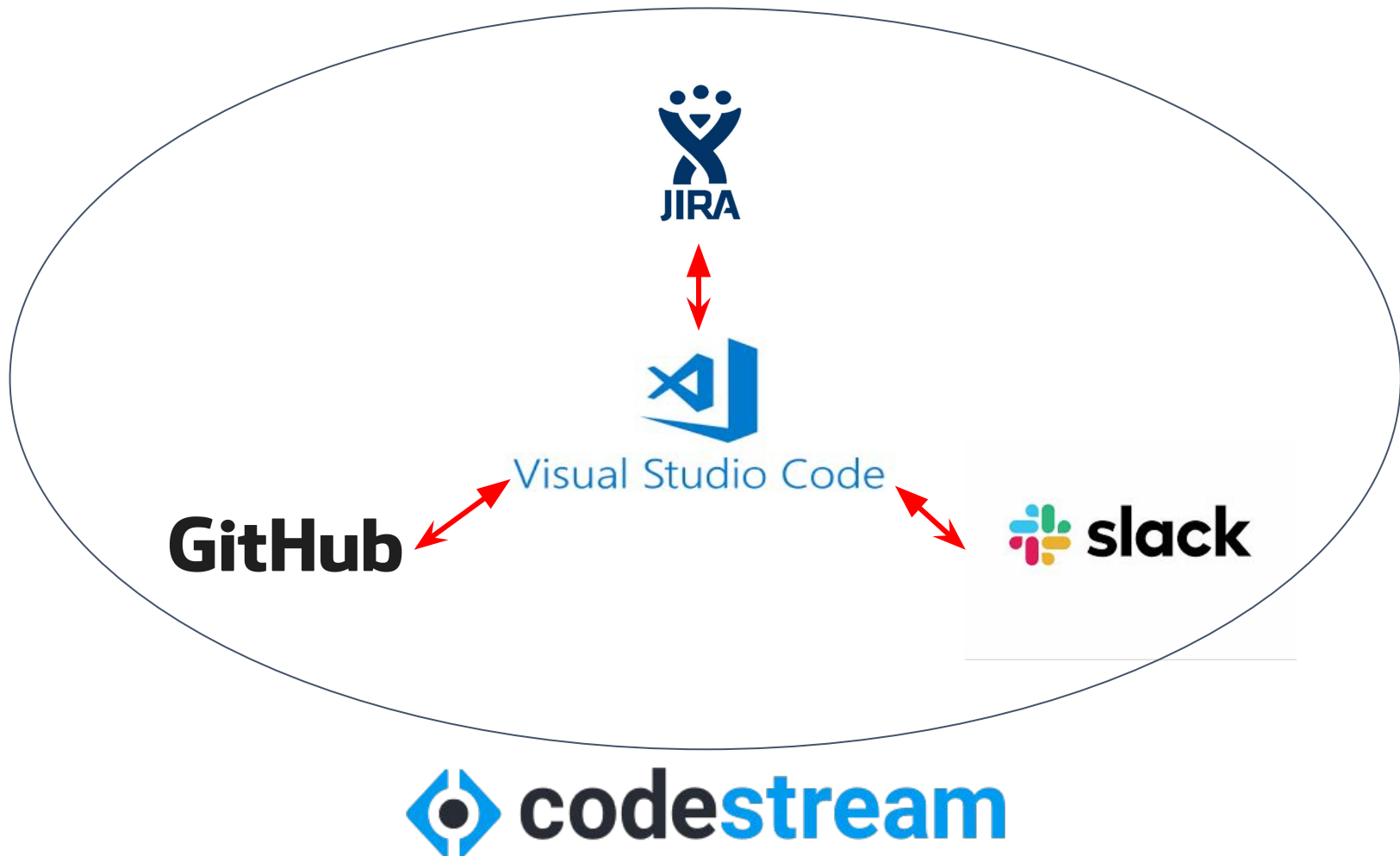
1. Recibir notificación en el editor, click.
2. Clic para hacer un pull & checkout a la rama.
3. Realizar la revisión de código en contexto y comentar en cualquier parte que te parezca, sin restricciones.
4. Terminar la revisión.

# Beneficios de la integración en el editor

- Ahorras tiempo.
- Reduces distracciones.
- Tienes acceso a todo el código en todo momento.
- Mejoras la calidad.
- Mejoras la comunicación.
- No tienes que cambiar de herramientas.

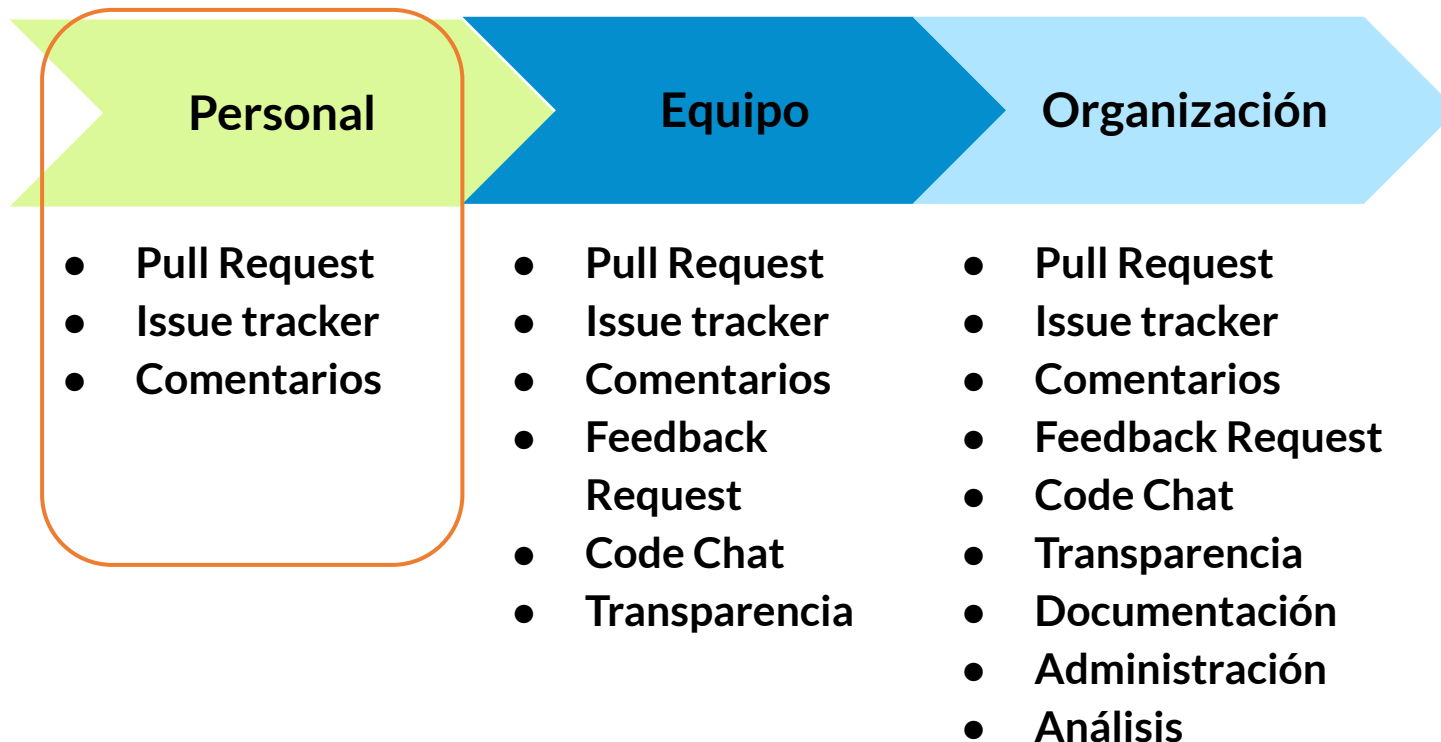


# Herramienta de integración



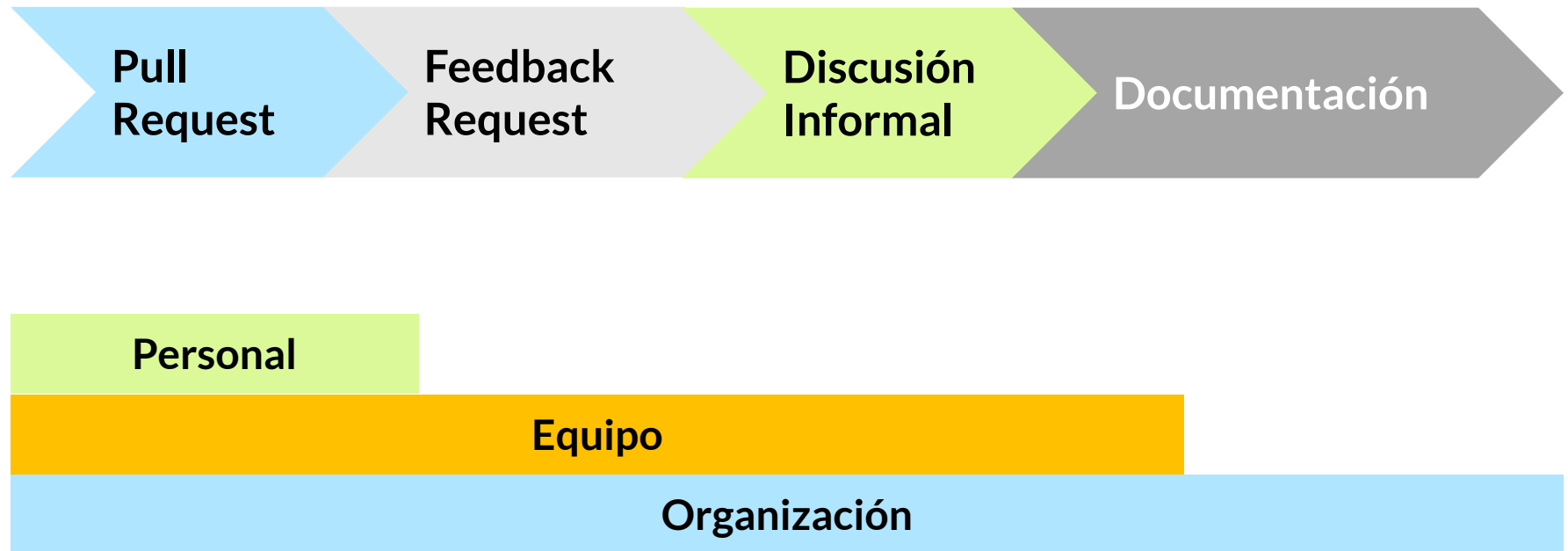
# Implementación del flujo moderno

## Uso/Efecto



# Evolución del flujo

## El camino hacia *Shift Left*



# Práctica I: instalación de CodeStream y GitHub en VS Code

---

---

# El flujo moderno: productividad

# Tradicional vs. Moderno

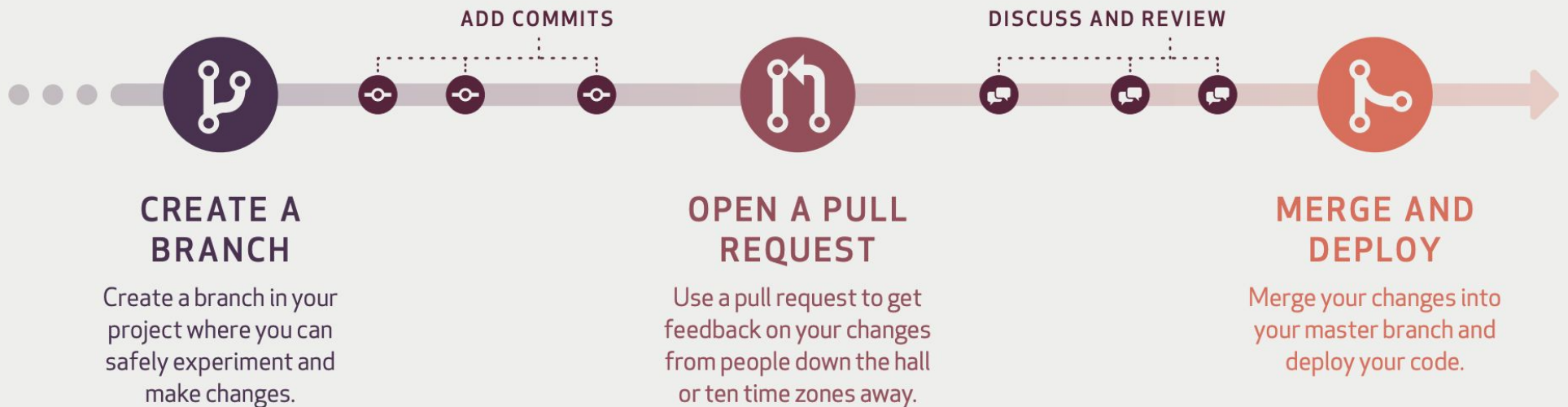
## ● Tradicional

- Herramientas no integradas.
- Revisión de código en el PR.
- Más líneas de comando.
- Cambio de contexto entre sitio web y tu editor.
- Comunicación externa al editor.
- Documentación externa al editor.

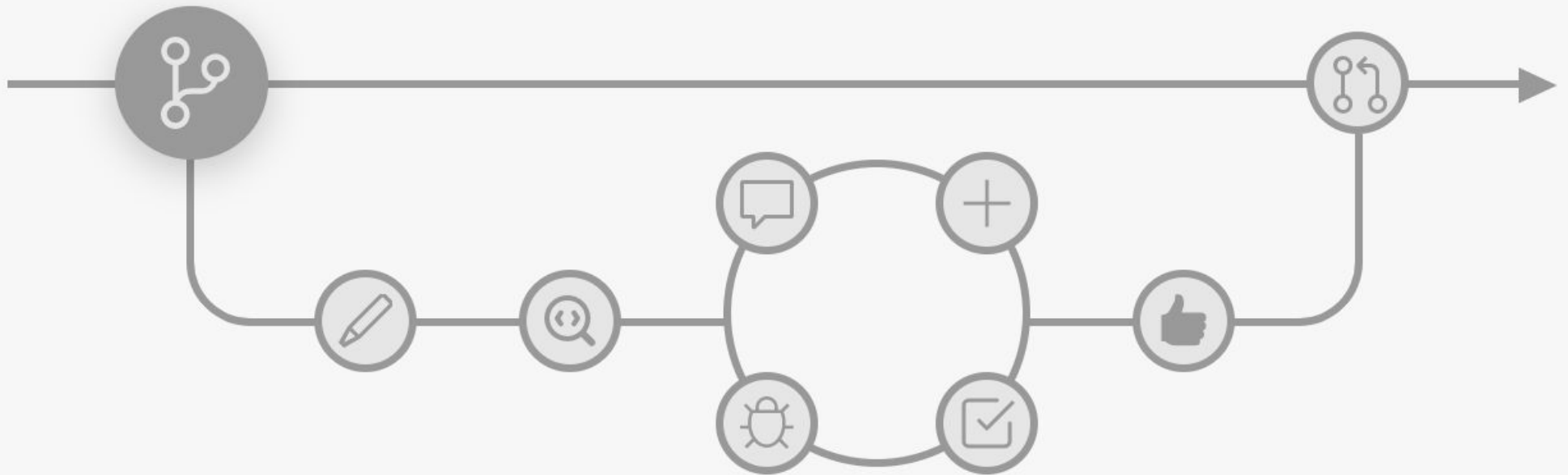
## ● Moderno

- Herramientas integradas en el editor.
- Revisión de código en el PR y en cualquier líneas de código.
- Revisión de código antes del commit.
- Más automatizado.
- Menos cambio de contexto.
- Comunicación integrada.
- Documentación integrada.

# Flujo Branch tradicional



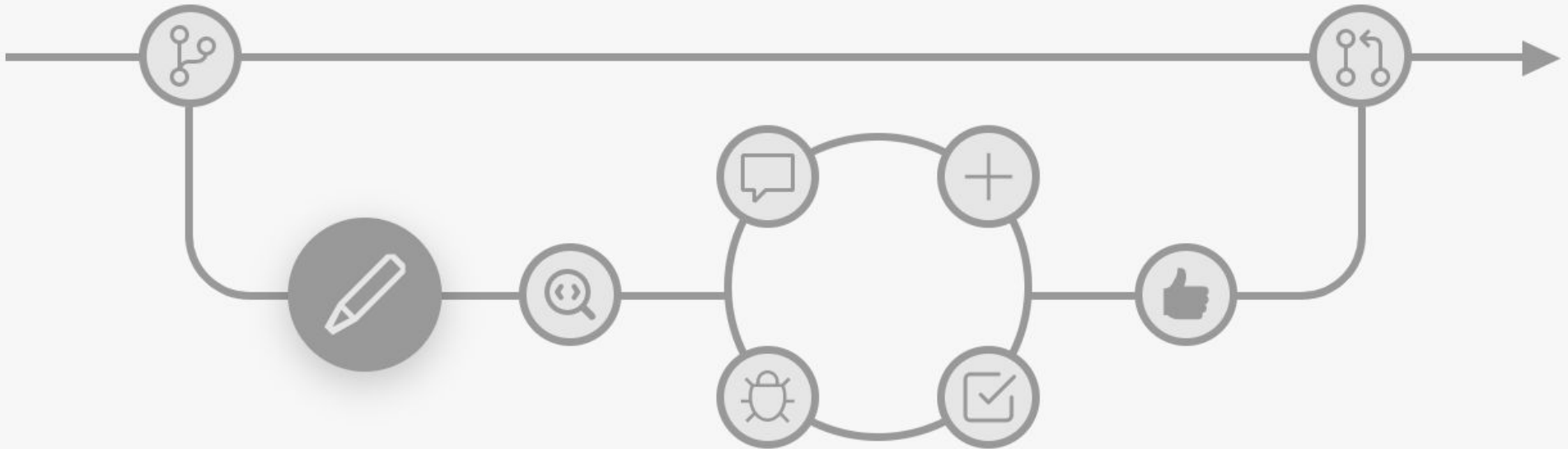
# Flujo Branch moderno



- a. Asignar ticket
- b. Crear branch
- c. Notificar al equipo

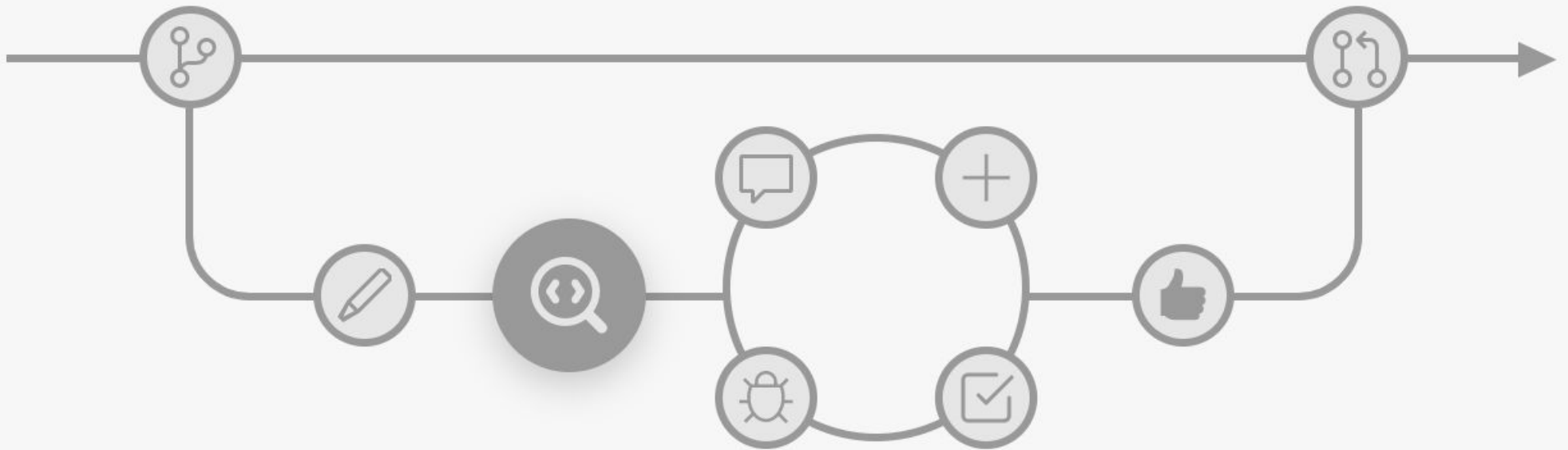


# Flujo Branch: escribir código



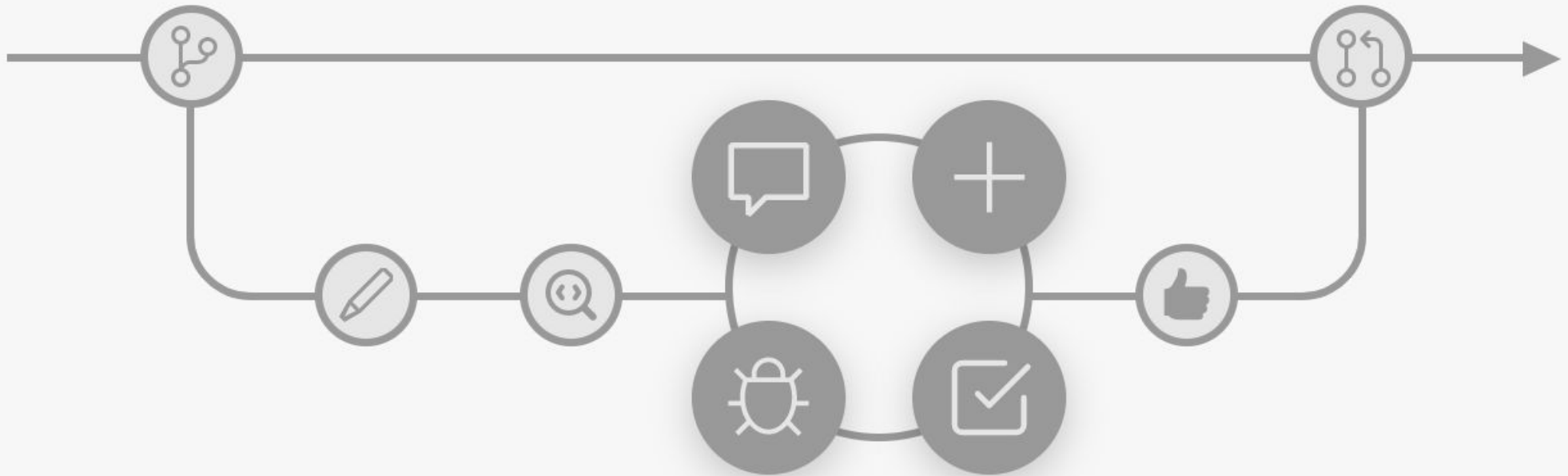
- a. Pedir comentarios
- b. Obtener sugerencias
- c. Explicar dirección

# Flujo Branch: sugerencias



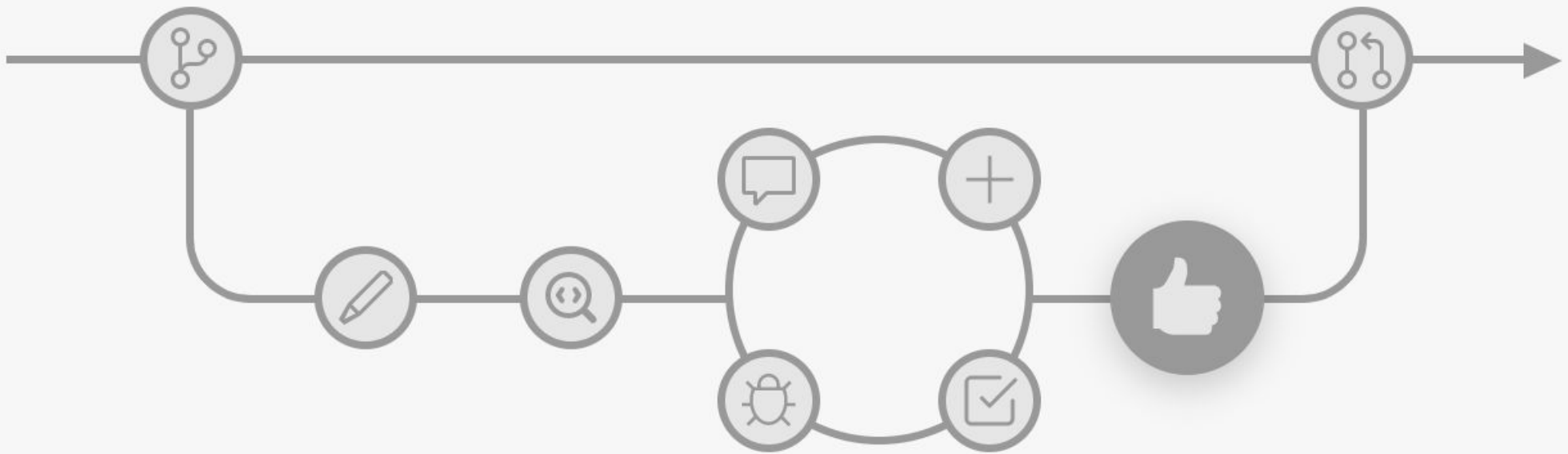
- a. Antes de un pull request
- b. Comunicación informal
- c. Documenta el proceso

# Flujo Branch: revisión de código pre-PR



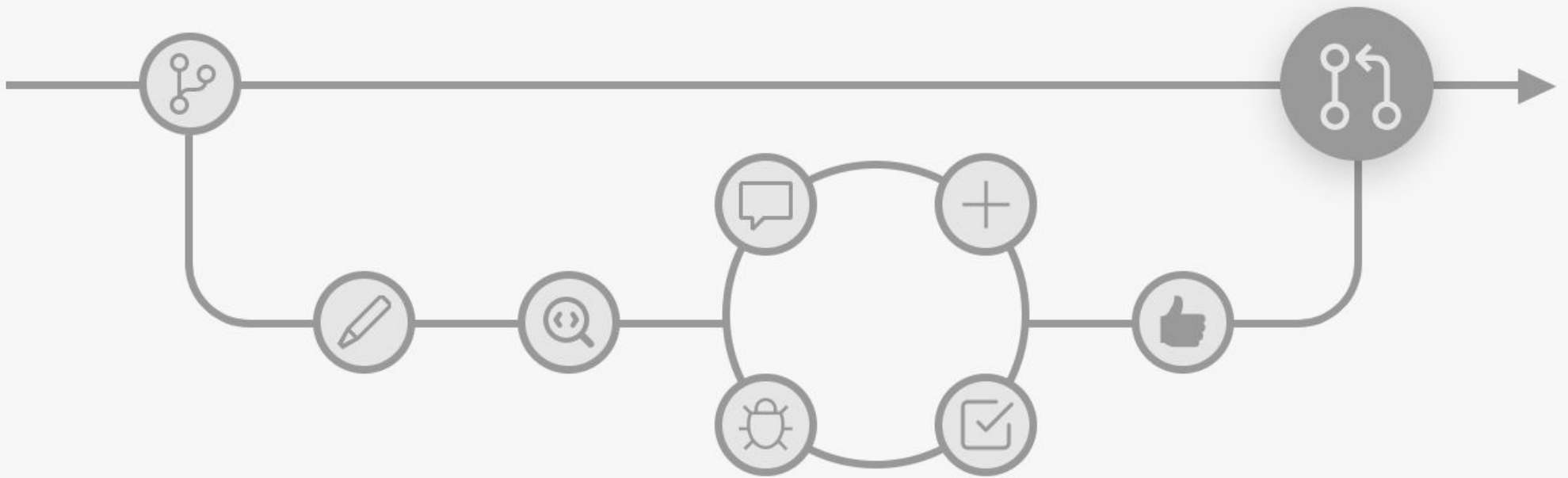
- a. Antes de un pull request
- b. Comunicación formal
- c. Se incorpora al PR

# Flujo Branch: aprobación final



- a. Antes de un pull request
- b. Aprobación formal
- c. Reducción de pasos

# Flujo Branch: PR y Merge



- a. 2 clics para un merge
- b. Incluye documentación
- c. Shift Left

# Pull Request o Feedback Request (*Shift Left*)

- **Flujo Personal:**
  - Pull Request en el editor.
  - Mejor Diff.
  - Comentarios adicionales.
  - Eficiencia administrativa.
- **Flujo Equipo:**
  - Metodología compartida.
  - Revisión de código atomizada.
  - Comentarios adicionales.
  - Más informal, más eficiente.



**Aumento en productividad**

**32%**



# GitHub en tu editor



# ¿Por qué integrar GitHub en tu editor?

- ¿Por qué trabajar en GitHub.com?
- ¿Por qué tener que saltar de GitHub.com a tu editor, ida y vuelta, para entender el código?
- ¿Por qué se puede comentar solamente sobre los cambios en este PR?
- ¿Por qué comparar archivos dentro de un sitio web en vez del editor?



# Beneficios de la integración (1)

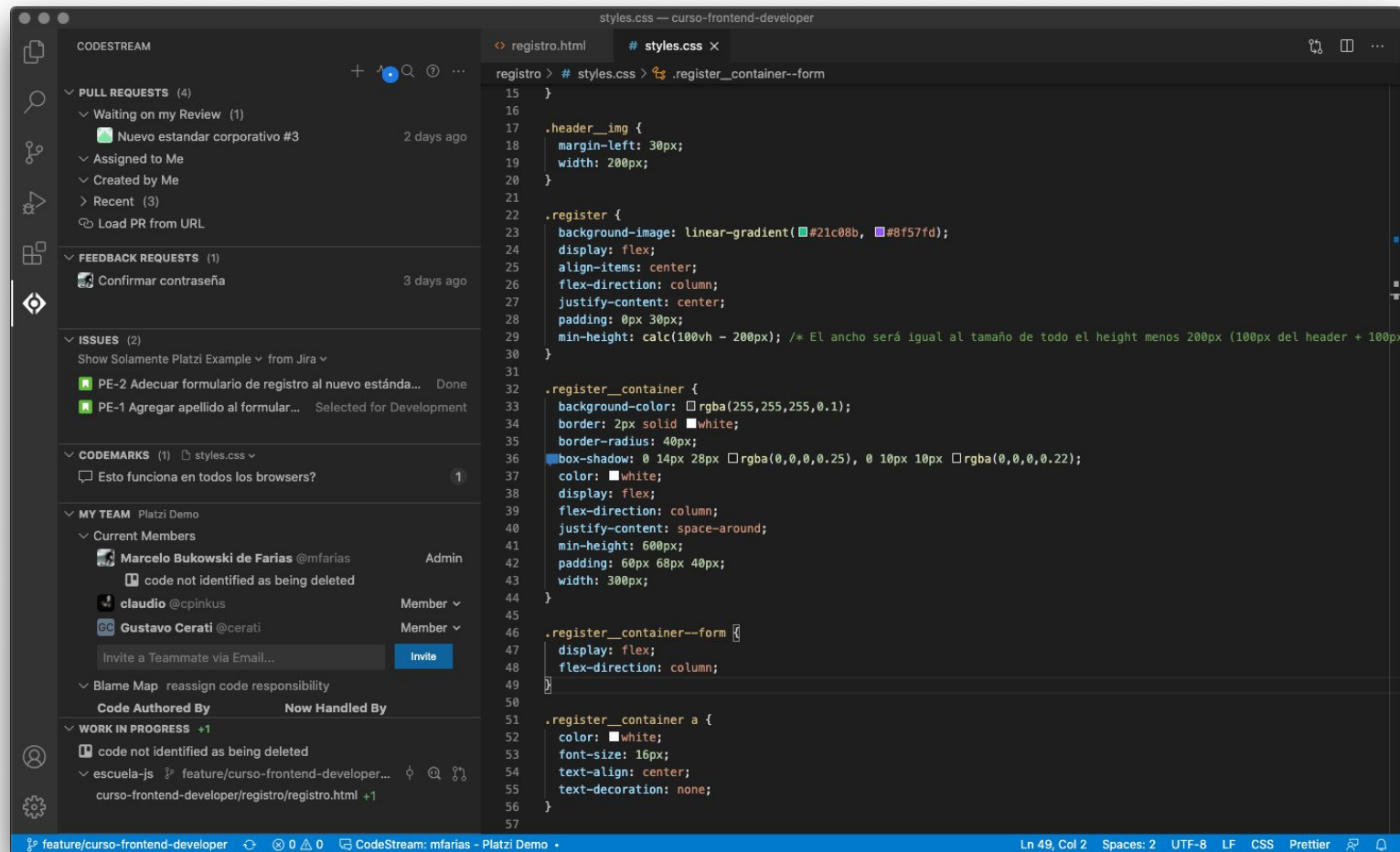
- Eliminación del cambio de contexto.
- Check out a una branch en un solo clic.
- Ejecutar una compilación durante la revisión.
- Saltar a la definición.
- Tus atajos de teclado, temas y personalizaciones preferidos.



# Beneficios de la integración (2)

- La herramienta “diff” nativa de tu editor.
- El contexto completo de tu repositorio.
- Agregar comentarios en cualquier parte del repositorio relacionados con cualquier revisión de código.

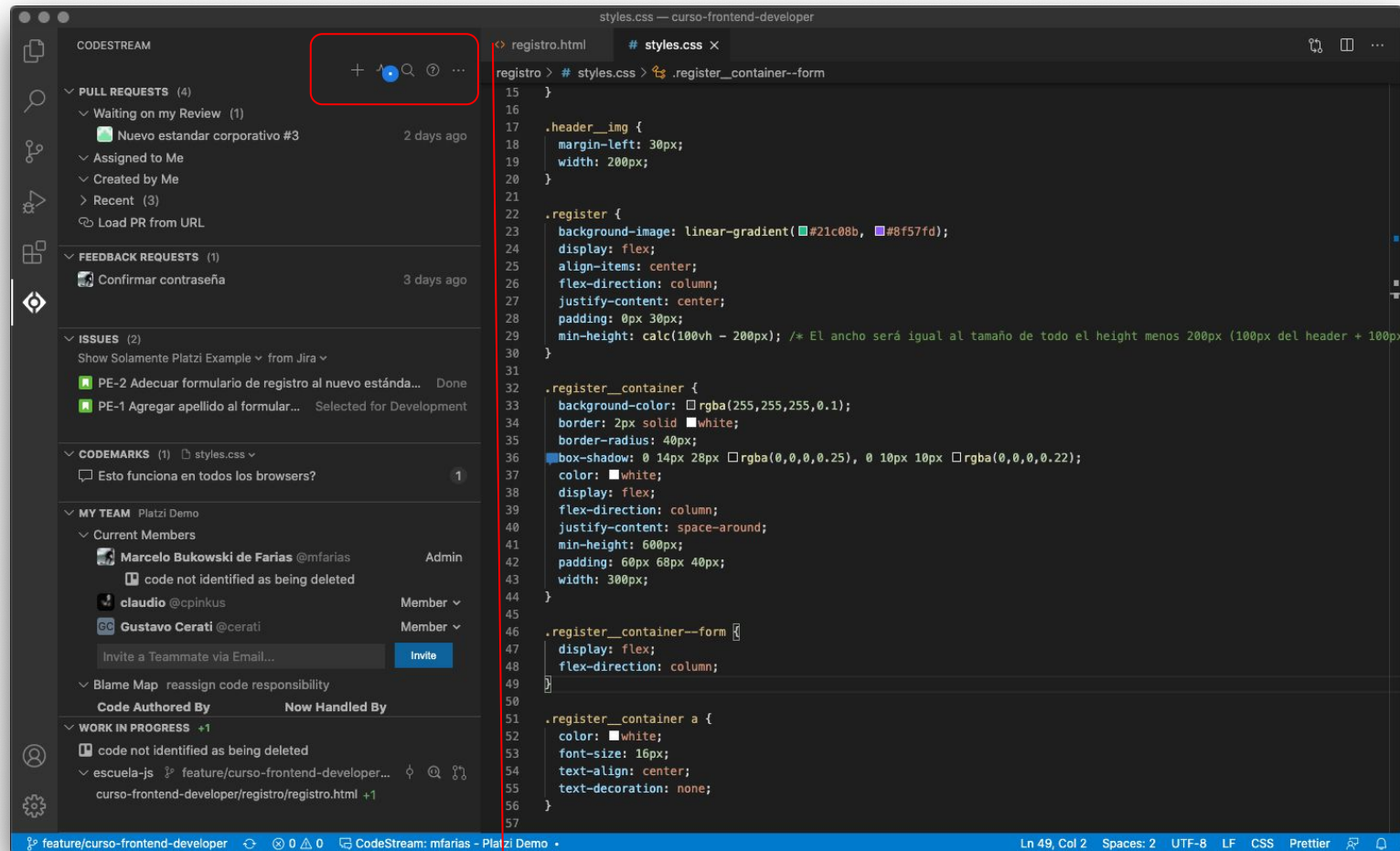
# Navegación: CodeStream



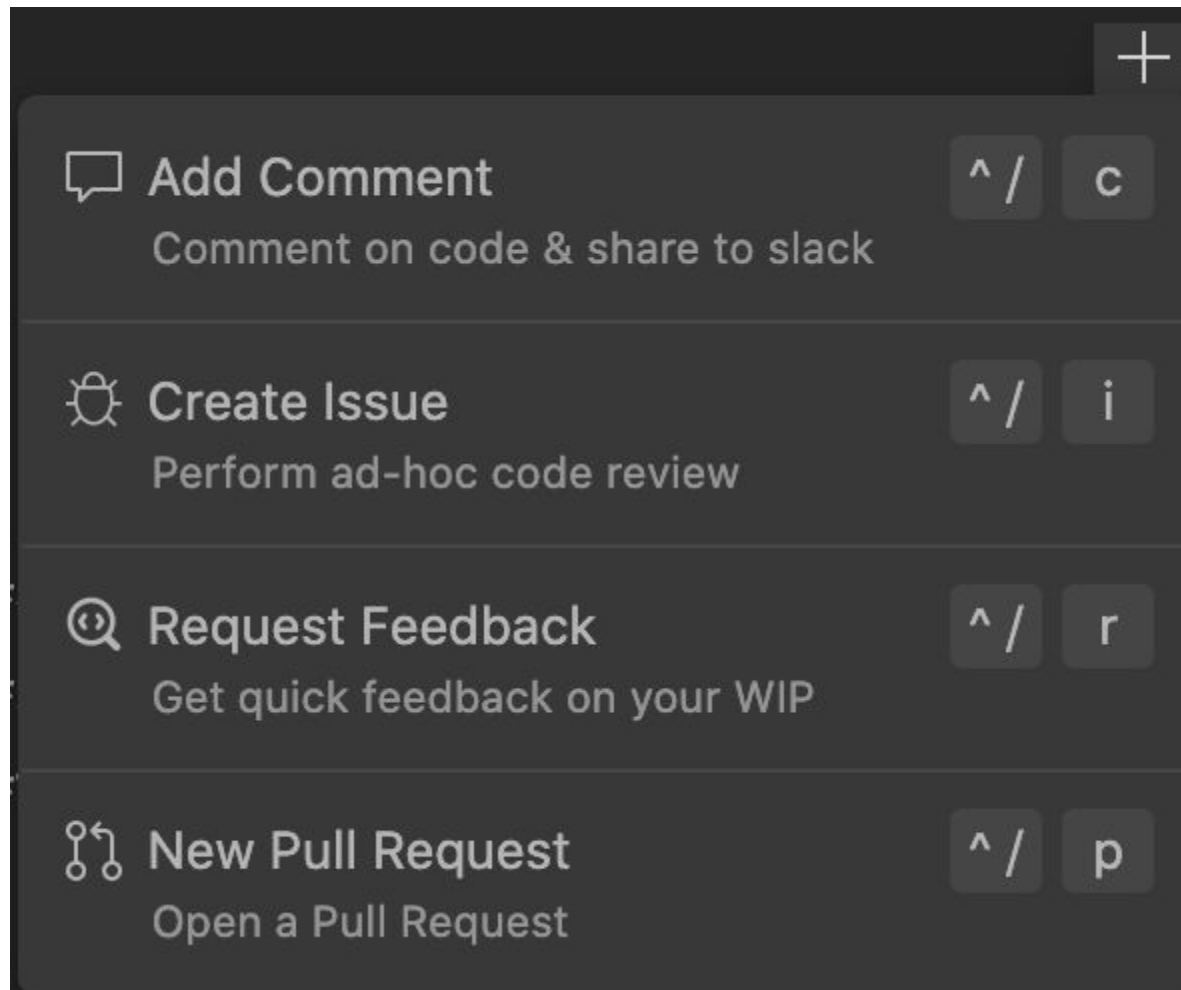
# Navegación: Controles

2 3

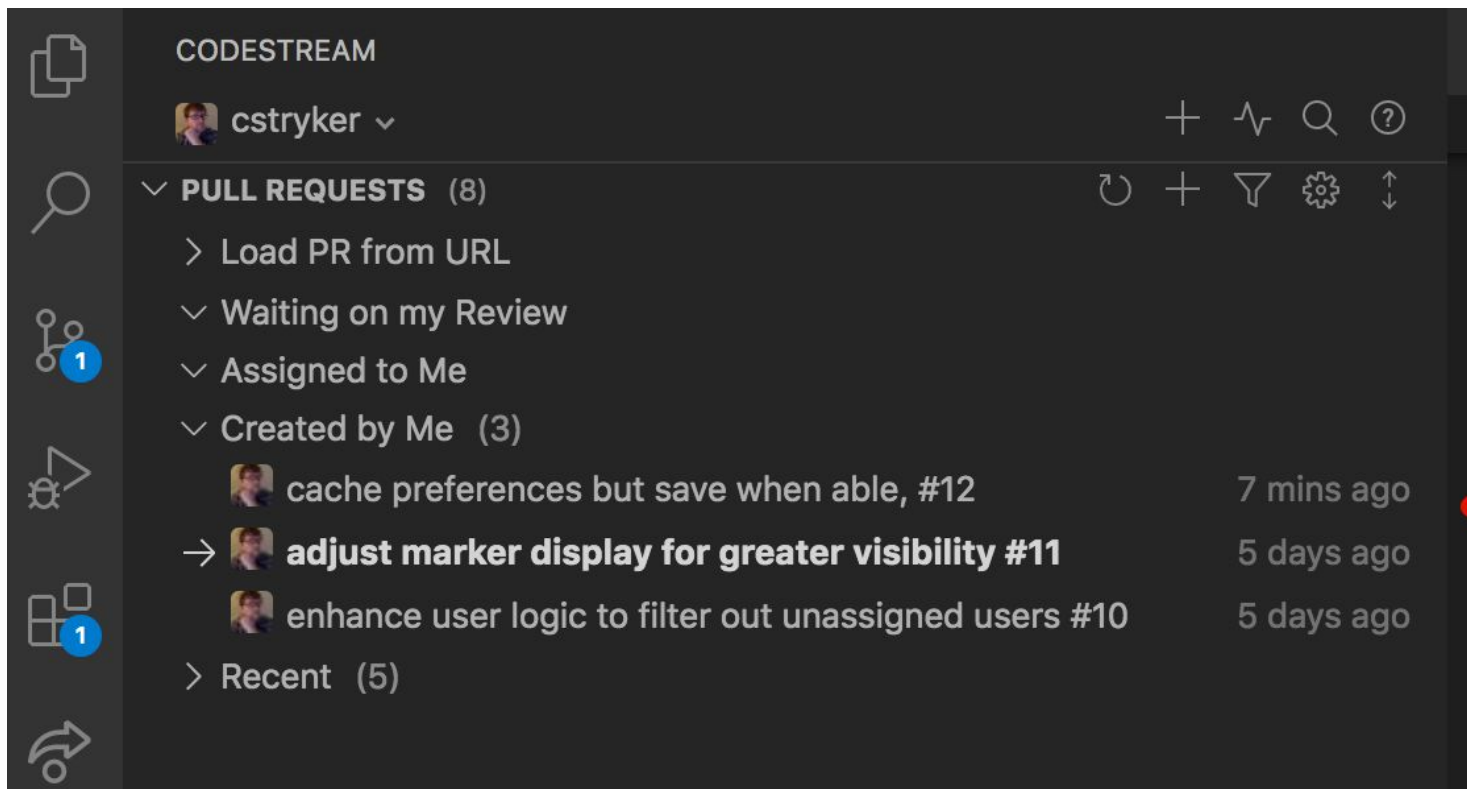
1



# Navegación: 4 funciones



# Navegación: Pull Requests



# Navegación

## ✓ PULL REQUESTS (5)

✓ Waiting on my Review

✓ Assigned to Me

✓ Created by Me

✓ Recent (5)



PE-3 Agregar fecha de nacimiento al formulario de registro #5...

22 hrs ago







# Eficiencia del PR en tu editor

**Desde el panel CodeStream puedes:**

- Agregar comentarios.
- Crear issues en Jira.
- Crear un Feedback Request.
- Crear un Pull Request.



# Eficiencia del PR en tu editor

- Puedes comentar sobre cualquier línea de código.
- Puedes eliminar la ineficiencia administrativa.

# Práctica II: Pull Request integrado en VS Code



---

# Feedback Request y gestión de tareas (JIRA)

# Feedback Requests (CodeStream)

- Relacionado con el concepto de un Pull Request.
- Se realiza antes en el flujo para poder atomizar la colaboración.
- Permite pedir feedback, es decir, comentarios sobre código en cualquier estado de tu repositorio.
- Es mucho más fácil para el revisor.

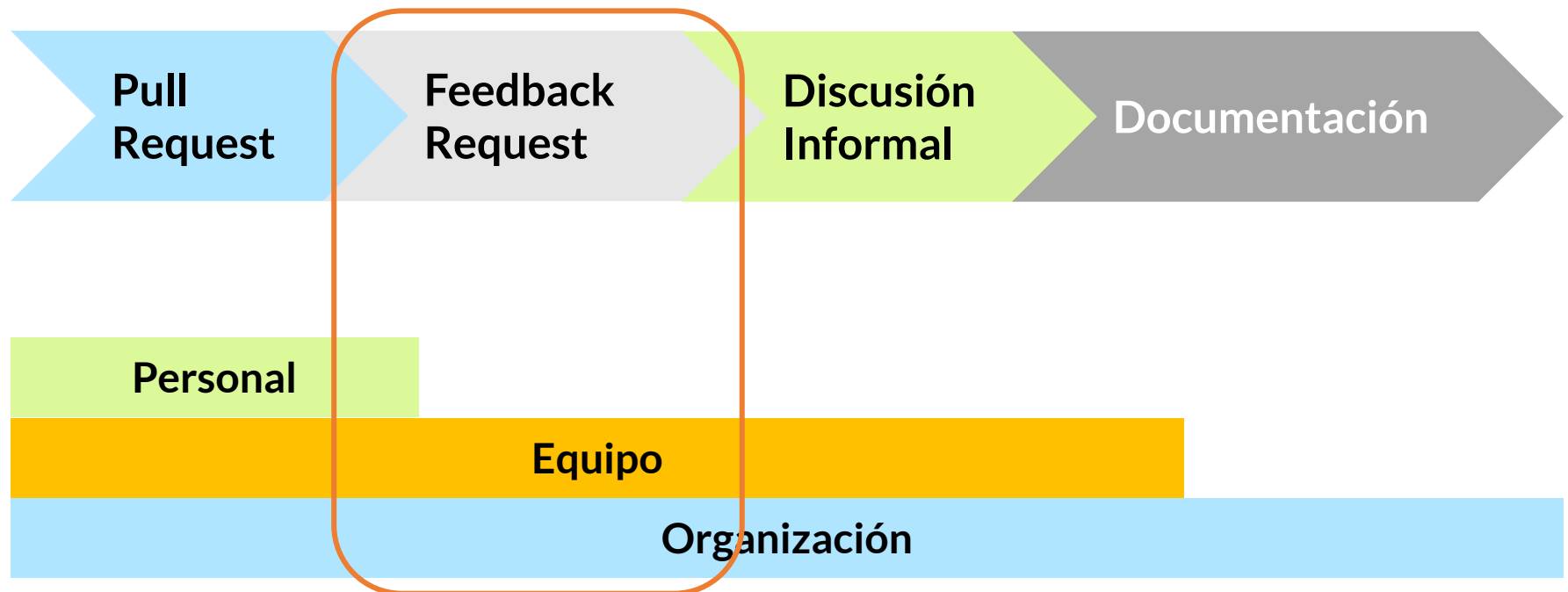


# Principios de colaboración Shift Left

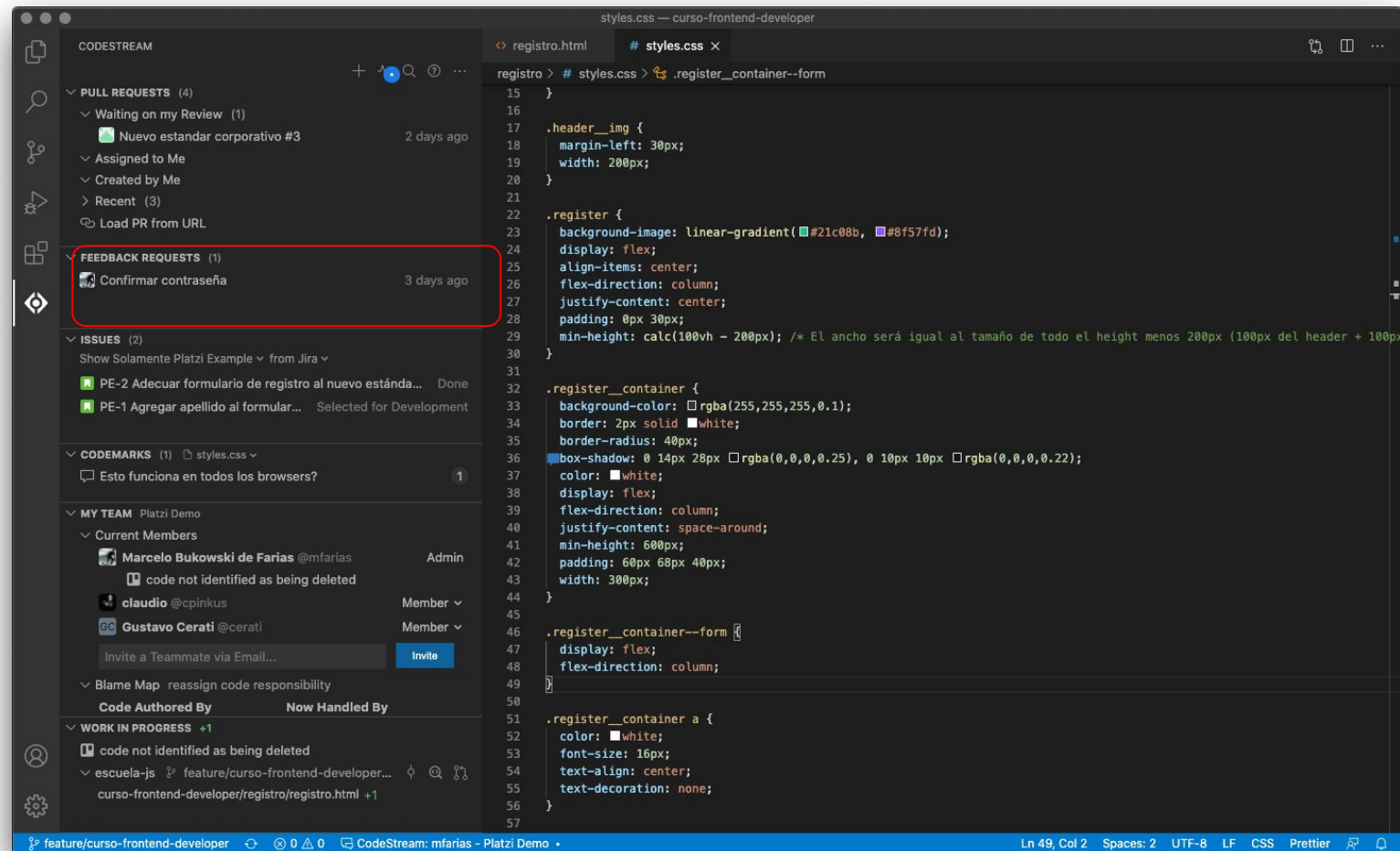
1. Antes mejor que después.
2. Colaboración atomizada.
3. Más consulta que aprobación.
4. Reducir fricción administrativa.
5. Compartir conocimientos.

# Evolución del flujo

## El camino hacia *Shift Left*



# Feedback Requests

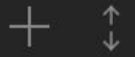




# Definición Feedback Requests


## ✓ FEEDBACK REQUESTS



Lightweight, pre-PR code review. Get quick feedback on any code, even pre-commit. [Learn more.](#)







Revisión de código ligera antes del pull request. Para recibir comentarios sobre cualquier código, incluso antes de un commit.

# Feedback Request: Secciones

1  **cpinkus** is requesting feedback in `escuela-js` on branch `feature/curso-frontend-developer`

2 Title  

3 Description (Optional)

4 REVIEWERS – ANYONE CAN APPROVE   
 sampol.90  cerati 

5 CHANGED FILES  
`curso-frontend-developer/not-found/not-found.html` +28  
`curso-frontend-developer/not-found/styles.css` +112  
`curso-frontend-developer/registro/styles.css` +1

6 CHANGES TO INCLUDE IN REVIEW  
☒ Saved Changes (Working Tree) 1 file

PUSHED COMMITS

<input checked="" type="checkbox"/> Added: Not found page	teffcode Jun 30, 2019	1a8e45b9
<input type="checkbox"/> Added: background in header	Added: Menu	20e6a104
<input type="checkbox"/> Added: Menu		4056b38a

# Integración de proyectos/tareas en VS Code

- Incluye funcionalidad completa.
- Tres propósitos:
  - Automatización
  - Uniformidad
  - Comunicación
- Permite integrar más de una aplicación en la misma lista de tareas.
  - Por ejemplo, Jira, Trello y GitHub Issues.

# Ejemplo 3: Asignar un ticket

Sin integración

7

Pasos

Con integración

2

Pasos

# Ejemplo 3: asignar un ticket

## Sin integración

1. Navegar a tu instalación de Atlassian y elegir JIRA
2. Aplicar filtro a tus tickets, para encontrar el deseado
3. Mover el ticket a “In Progress”
4. Escribir código
5. Commit, push, etc.
6. Cuando terminas el código, mover el ticket a “En revisión”
7. Cuando se termina la revisión, mover el ticket a “Terminado”

## Con integración

1. Seleccionar un ticket en tu editor
2. Un solo clic para crear un branch, mover el ticket a “In Progress” y actualizar tu estado en Slack. Eso es todo

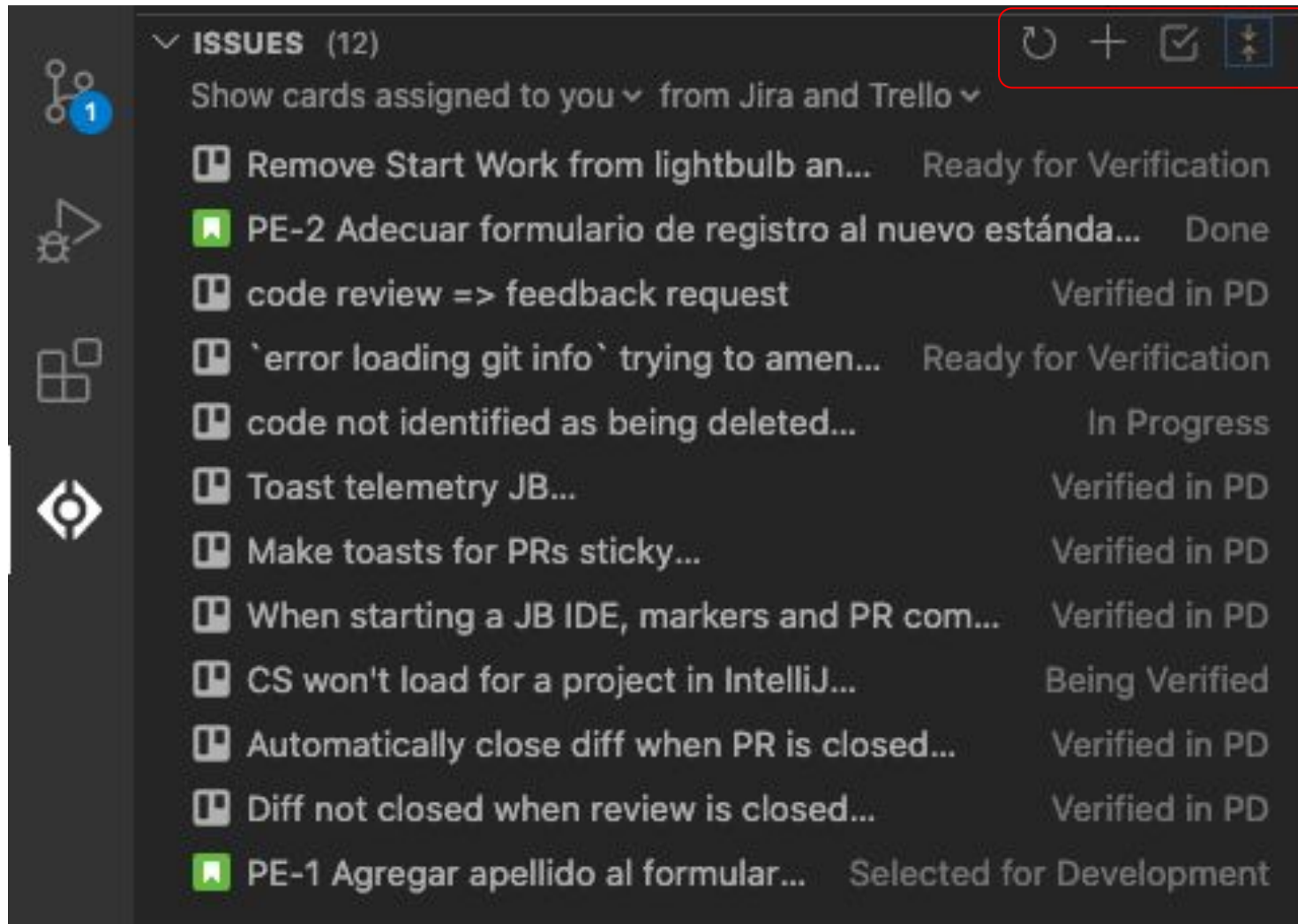
# Selección de una tarea

ISSUES (12)

Show cards assigned to you ▾ from Jira and Trello ▾

Remove Start Work from lightbulb an...	Ready for Verification
PE-2 Adecuar formulario de registro al nuevo estándar...	Done
code review => feedback request	Verified in PD
`error loading git info` trying to amen...	Ready for Verification
code not identified as being deleted...	In Progress
Toast telemetry JB...	Verified in PD
Make toasts for PRs sticky...	Verified in PD
When starting a JB IDE, markers and PR com...	Verified in PD
CS won't load for a project in IntelliJ...	Being Verified
Automatically close diff when PR is closed...	Verified in PD
Diff not closed when review is closed...	Verified in PD
PE-1 Agregar apellido al formular...	Selected for Development

# Selección de una tarea



The screenshot displays a task management interface with a dark theme. On the left is a sidebar with icons for a network diagram, a gear, a grid, and a target. The main area is titled 'ISSUES (12)' and shows a list of tasks. A red box highlights the top right of the list, containing icons for refresh, add, check, and a vertical ellipsis. The tasks are listed with their titles and status on the right. The task 'PE-1 Agregar apellido al formular...' is highlighted with a green icon and labeled 'Selected for Development'.

Task Title	Status
Remove Start Work from lightbulb an...	Ready for Verification
PE-2 Adecuar formulario de registro al nuevo estándar...	Done
code review => feedback request	Verified in PD
`error loading git info` trying to amen...	Ready for Verification
code not identified as being deleted...	In Progress
Toast telemetry JB...	Verified in PD
Make toasts for PRs sticky...	Verified in PD
When starting a JB IDE, markers and PR com...	Verified in PD
CS won't load for a project in IntelliJ...	Being Verified
Automatically close diff when PR is closed...	Verified in PD
Diff not closed when review is closed...	Verified in PD
PE-1 Agregar apellido al formular...	Selected for Development

# Filtros de JIRA integrados

×

## Create a Custom Filter

Example: assignee=currentuser() AND status!=Closed

See [Jira Advanced Searching](#) for documentation on JQL.



# Selección de un ticket

1

Error in blame logic

2

@pez: I'm seeing a lot of errors like the one attached in some customer logs. Do you spot anything wrong with this logic?

[codestream] shared/agent/src/git/gitService.ts (Lines 1245-1248)


3

```
01.      patch.hunks.forEach(hunk => {  
02.          const oldEnd = hunk.oldStart + hunk.oldLines;  
03.          options.push(`-L${hunk.oldStart},${oldEnd}`);  
04.      });
```

4

☒ Move this card to In Progress ▾

5

☒ Set up a branch in  codestream ▾



develop ▾

feature/7tmnXzmU-error-in-blame-logic ▾

6

☒ Update my status on Slack

Cancel

Create Branch & Start Work

# Ventajas de tener un issue tracker integrado

- Agregar un ticket mientras escribes o revisas código.
- Conectar el ticket directamente al código.
- Notificar a la persona indicada que hay un ticket y dirigirlo al lugar correcto.
- No tener que cambiar de aplicación o contexto.
- Crear un registro de los tickets asociados al código mismo.

# Práctica III: Flujo Tradicional Pull Request



# Práctica III: Flujo Moderno Pull Request



# Práctica III: Flujo Moderno Feedback Request

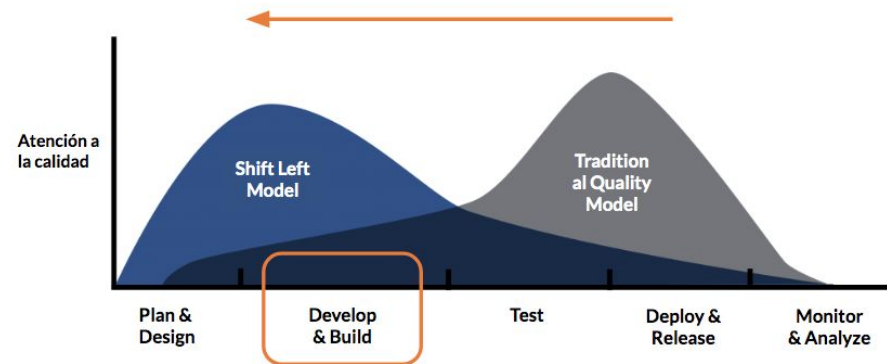


---

# Colaboración y comunicación (Code Chat, Slack)

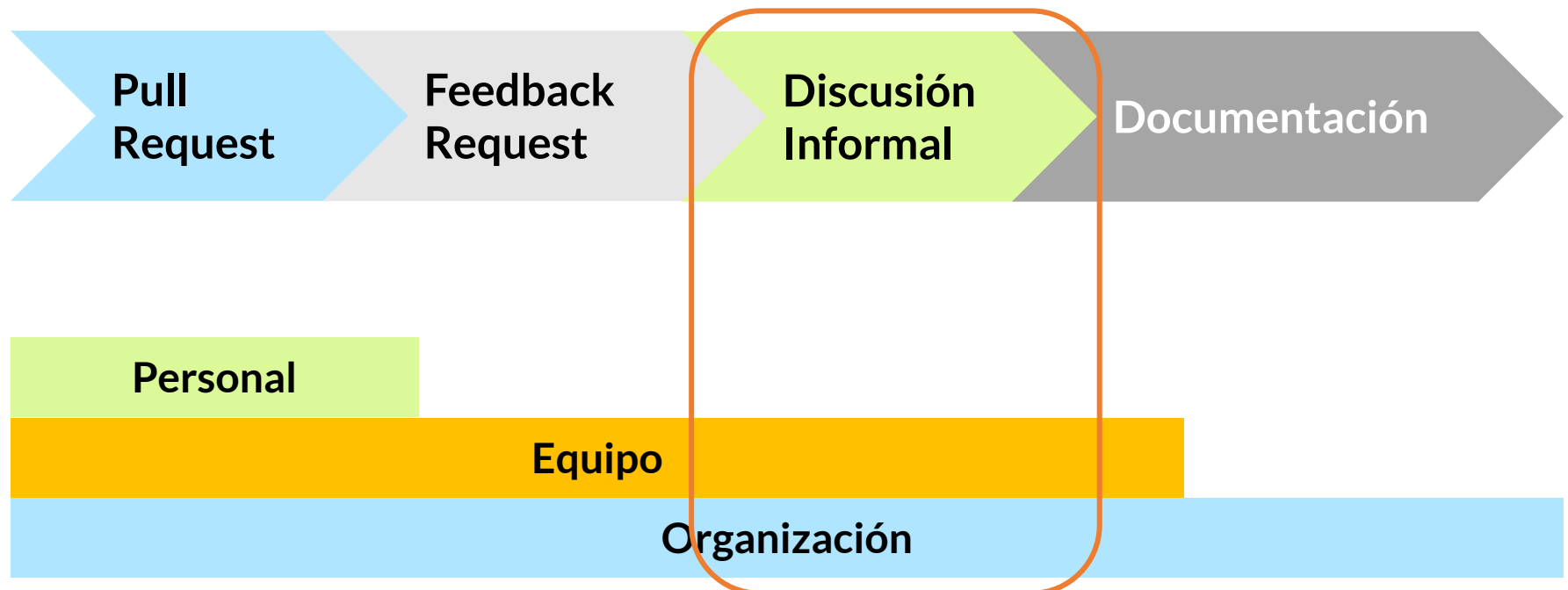
# Principios de colaboración Shift Left

1. Mejor antes que después
2. Colaboración atomizada
3. Más consulta que aprobación
4. Reducir fricción administrativa
5. Compartir conocimientos



# Evolución del Flujo

## El camino hacia *Shift Left*





## Ejemplo 4: Pedir ayuda

Sin integración

18

Pasos

Con integración

2

Pasos

# Ejemplo 4: pedir ayuda en Slack

## Sin integración

1. Select bloque de código
2. Copiar
3. Alt-tab a Slack
4. Elegir el canal
5. Teclear ``` pegar texto y ``` de vuelta
6. Explicar el contexto
7. Dar detalles del repo y proyecto
8. Teclear path y archivo fuente
9. Copiar número de línea
10. Hacer referencia a la función(es)
11. Encontrar a la persona correcta
12. Alt-tab a Terminal
13. cd al directorio correcto
14. git blame | grep
15. Alt-tab nuevamente a Slack
16. Copiar/pegar email del autor, email para mencionar
17. Escribir la pregunta (finalmente)
18. Alt-tab de vuelta al editor

## Con integración

1. Seleccionar el bloque código
2. Hacer la pregunta



# Code Chat

- El Code Chat es mensajería de equipo diseñada para trabajar con líneas y bloques de código.
- Detecta cambios y diferencias en distintas versiones del mismo bloque.
- Contiene la meta-información para evolucionar con el código.
- Se integra con Slack, Pull Requests, Jira.
- Se transforma en documentación.



# Propósito del Code chat

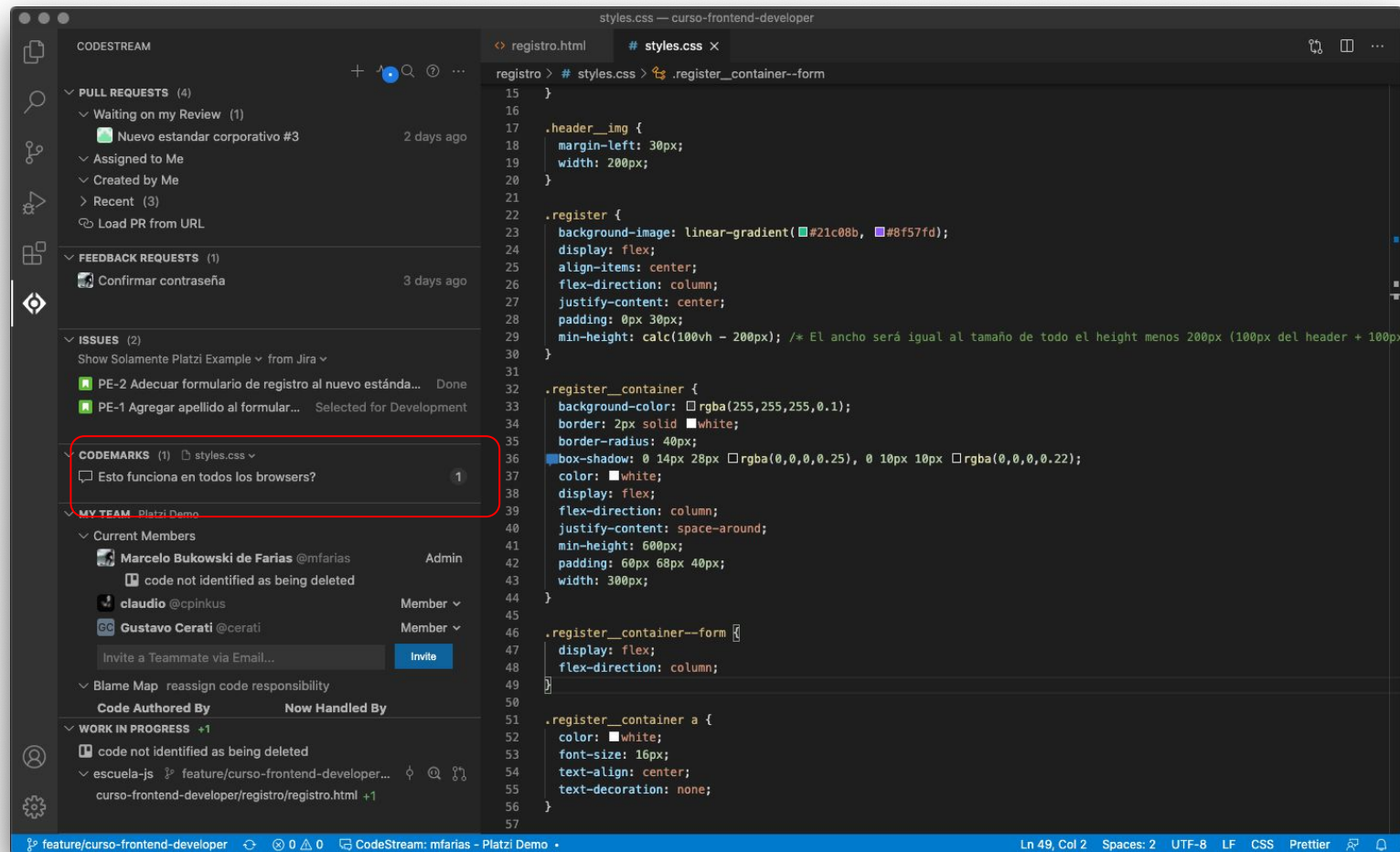
- Colaboración informal atomizada.
- Permite hacer preguntas y sugerencias sobre cualquier parte del código.
- Conecta distintas partes del flujo.
- Conecta distintos bloques de código.
- Documenta el código.
- Explica decisiones ya tomadas.



# Codemarks

- Cada vez que se crea una unidad de comunicación en CodeStream se crea un “codemark”.
- Un codemark es un enlace entre la información sobre el código (metadata) y el bloque de código al que se refiere.
- Un codemark puede ser un mensaje, un issue o un permalink (enlace permanente).
- Codemarks son exportables.

# Sección Codemarks




# Sección Codemarks

▼ CODEMARKS (3)  styles.css ▼



 Corregir padding


 Esto funciona en todos los browsers?

1


 @cerati Por qué 600?

1

# Codemark Abierto


 **mfarias** 11 days ago

Esto funciona en todos los browsers?

 curso-frontend-developer/registro/styles.css

```
36. box-shadow: 0 14px 28px rgba(0,0,0,0.25), 0 10px 10px rgba(0,0,0,0.22);
```




ACTIVITY

 **cpinkus** 11 days ago (edited)

IE 11 suprime box-shadow en tables con border-collapse: collapse. Salvo eso, todo bien.

ADD REPLY

Reply...

Submit

WORK IN PROGRESS

```
19 width: 200px;
20 }
21
22 .register {
23   background-image: linear-gradient(#21c08b, #8f57fd);
24   display: flex;
25   align-items: center;
26   flex-direction: column;
27   justify-content: center;
28   padding: 0px 30px;
29   min-height: calc(100vh - 200px); /* El ancho será igual al tamaño de todo el
30 }
31
32 .register__container {
33   background-color: rgba(255,255,255,0.1);
34   border: 2px solid white;
35   border-radius: 40px;
36   box-shadow: 0 14px 28px rgba(0,0,0,0.25), 0 10px 10px rgba(0,0,0,0.22);
37   color: white;
38   display: flex;
```



# Detalle: Codemark Abierto

 **mfarias** 11 days ago

Esto funciona en todos los browsers?

 curso-frontend-developer/registro/styles.css

```
36. box-shadow: 0 14px 28px rgba(0,0,0,0.25), 0 10px 10px rgba(0,0,0,0.22);
```

## ACTIVITY

 **cpinkus** 11 days ago (edited)

IE 11 suprime box-shadow en tables con border-collapse:collapse. Salvo eso, todo bien.

## ADD REPLY

Reply...




Submit

# Sección Codemarks

▼ CODEMARKS (3)  styles.css ▼

 Corregir padding

 Esto funciona en todos los browsers?

 @cerati Por qué 600?



1

1

# Spatial View

 mfarias 5 days ago

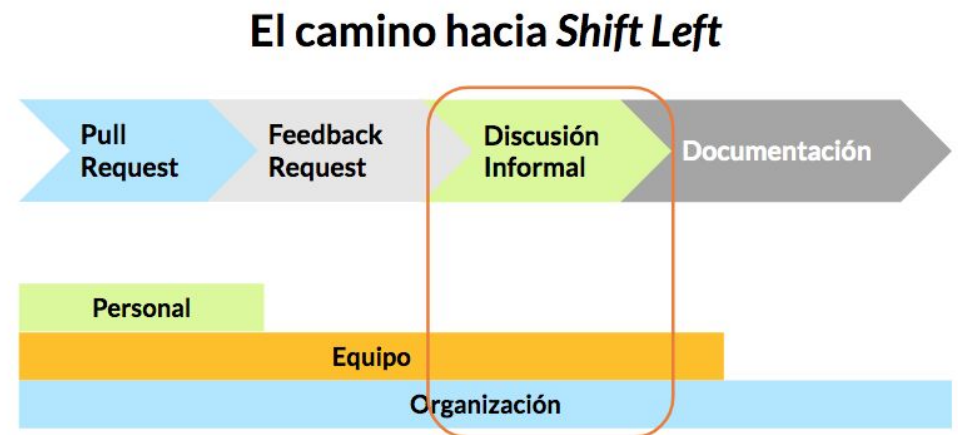
Esto funciona en todos los browsers?

  1

```
31
32 .register__container {
33   background-color: rgba(255,255,255,0.1);
34   border: 2px solid white;
35   border-radius: 40px;
36   box-shadow: 0 14px 28px rgba(0,0,0,0.25), 0 10px 10px rgba(0,0,0,0.22);
37   color: white;
38   display: flex;
39   flex-direction: column;
40   justify-content: space-around;
41   min-height: 600px;
```

# Code Chat: resumen

- El code chat facilita la colaboración informal.
- Se integra con los sistemas de comunicación existentes.
- Se adecúa a la evolución y las diferencias del código.
- Se utiliza en cualquier parte del repositorio.



# Comunicación dentro del editor con Code Chat y Codemarks



---

# Trabajo remoto y transparencia

# El trabajo remoto

- La nueva norma es el trabajo remoto.
- Muchas empresas están decidiendo hacer la transición permanente.
- Se terminó la comunicación informal en persona.
- ¿Qué herramientas hacen falta para adaptarse?

[Artículo sobre la transparencia](#)

# El flujo moderno: transparencia

- Una filosofía de trabajo.
- Estar abierto a compartir conocimiento.
- Estar abierto a mostrar trabajo con defectos.
- Saber qué está haciendo el equipo.
- Mostrar lo que estás haciendo tú.



# De transparencia a visibilidad

- La visibilidad es la implementación de la transparencia
- Tecnologías que permiten tener visibilidad
  - Calendario
  - Status en Slack/MS Teams/etc.
  - Zoom Call
- Live View en CodeStream

# Tu equipo

1

2


3


4


5

▼ MY TEAM Platzi Demo


▼ Current Members



 **claudio** @cpinkus

 **Gustavo Cerati** @cerati

 **Marcelo Bukowski de Farias** @mfarias

Admin

 Pointer to current file and status of visited files not updated when using native diff navigation in PRs

 escuela-js  feature/ticket-PE-4-pe-4-agregar-ciudad- +1

Invite a Teammate via Email...

Invite


▼ Blame Map reassign code responsibility


**Code Authored By**

tefy.1@live.com

eaguilar@buildingengines.com


**Now Handled By**






 cerati

 cerati



# Live View

1



**Dave Hersh** @dave  Admin

-  Line breaks displayed in Change Requests section
-  hello-bronx  feature/foo +3 -1
-  hello-ny  feature/ticket-highlight +2




**Brett Paden** @paden Member ▾

-  codestream-server  develop +1 -1

**Jeff Strinko** @strinko Member ▾

-  codestream  develop +1 -1

**Larry Wahl** @larry Member ▾

-  hello-ny  feature/ticket-highlight +2 @2 

2

# Potencial conflicto de Merge



# Resumen: trabajo remoto y transparencia

- La implementación de una filosofía de transparencia es importante en el flujo moderno.
- El trabajo remoto es la nueva norma.
- Crear visibilidad con las herramientas de trabajo.
- Live View está integrado en CodeStream.

---

# La documentación y el flujo moderno

# El problema de la documentación

- La mayoría de los equipos de desarrollo no documentan sus proyectos.
- Menos el 20% de los desarrolladores usan la documentación interna para resolver preguntas.
- Al empezar un trabajo nuevo, los desarrolladores pasan el 75% del tiempo estudiando el código.
- Nadie sabe de antemano qué pregunta tendrá el otro.

# La documentación es parte del flujo moderno

- El flujo moderno apunta a la eficiencia.
- Muchos líderes de desarrollo responden a las mismas preguntas una y otra vez.
- Al usar Slack o MS Teams se pierde el contexto y el contenido.
- Al integrar el Code Chat se mantiene el contexto y el contenido en el código.
- El capturar y preservar el conocimiento es parte del activo (asset) de la organización.





# Qué debe capturarse

- Todas las actividades relacionadas con la base de código de la organización:
  - Comentarios
  - Mensajes
  - Issues
  - Errores en producción
  - Sugerencias
  - Diagramas
  - Frecuencia de colaboración
- Todos los metadatos relacionados con el código

# Donde debe vivir la documentación

- Con el código mismo:
  - Utilidad
  - Conectividad
  - Accesibilidad
  - Interactividad
- Debe ser exportable a otros formatos:
  - Sistema de documentación
  - Sistemas analíticos



# Documentación On Demand

1. En lugar de pensar qué documentar, fomentar las preguntas.
2. En lugar de esperar al pull request, fomentar las sugerencias.
3. En lugar de armar documentos de inducción (onboarding), dejar que el nuevo desarrollador explique lo que necesita.

# Menu de Codemarks

The screenshot displays a social media post by user 'mfarias' from 'yesterday'. The post content is '@cerati Por qué 600?' and includes a Google Card (GC) with the text 'Es lo llamado "typical breakpoint" en responsive design. Vea' and a link to 'https://www.w3schools.com/css/css\_rwd\_intro.asp'. A context menu is open over the post, listing the following actions: 1 Share, 2 Follow, 3 Copy link, 4 Archive, 5 Inject as Inline Comment, and 6 Reposition Codemark. To the right, a code editor shows CSS code for a media query: '@media only screen and (max-wi', '.register\_\_container {', 'background-color: transparent', 'border: none;', 'box-shadow: none;', 'padding: 0px;', 'width: 100%;', and '}'.

mfarias yesterday

@cerati Por qué 600?


☆ GC Es lo llamado "typical breakpoint" en responsive design. Vea  
[https://www.w3schools.com/css/css\\_rwd\\_intro.asp](https://www.w3schools.com/css/css_rwd_intro.asp)

1 2 3 4 5 6






Share  
Follow  
Copy link  
Archive  
Inject as Inline Comment  
Reposition Codemark




```
113 }  
114  
115 @media only screen and (max-wi  
116 .register__container {  
117     background-color: transparent  
118     border: none;  
119     box-shadow: none;  
120     padding: 0px;  
121     width: 100%;  
122 }  
123
```




# Agregar bloque




 **cpinkus** comment in 2 locations

@cerati:



 styles.css  feature/curso-frontend-developer  ce314fe

29. `min-height: calc(100vh - 200px);`   

 styles.css  feature/curso-frontend-developer  ce314fe

41. `min-height: 600px;` Cancel OK

+ Add Code Block

Share on  Slack or  MS Teams

Cancel Submit

# Agregar Tags

The image shows a dark-themed code editor interface. At the top, a comment by 'cpinkus' is shown for 'styles.css (Line 9)'. Below this, a code block is visible with the text '09. color: white;'. The interface includes options to 'Add Code Block', 'Send to tefy.1@live.com', and 'Share on Slack or MS Teams'. An 'Add Tag' dialog box is open in the foreground, featuring a text input field with the word 'Blanco', a 3x3 grid of color swatches (blue, green, yellow, orange, red, purple, cyan, grey, white), and 'Save' and 'Delete' buttons at the bottom.

cpinkus comment in styles.css (Line 9)

styles.css feature/curso-frontend-developer

09. color: white;

+ Add Code Block

Send to tefy.1@live.com ⓘ

Share on Slack or MS Teams


**Add Tag** ✕

Blanco

Blue	Green	Yellow
Orange	Red	Purple
Cyan	Grey	White ✓

Save Delete

# Ejemplo de Tag

 **cpinkus** comment in styles.css (Line 9)

Todo esto se refiere al blanco

M+ @ 😊 🎯 🏷

**TAGS**

Blanco

📄 styles.css 🔗 feature/curso-frontend-developer 🔑 ce314fe

09. color: white;🔗 📄 ✕

+ Add Code Block

☐ Send to tefy.1@live.com ⓘ

Share on 🗨 Slack or 👤 MS Teams

CancelSubmit

# Filtros y búsqueda

Filter & Search

Filters

Search comments, issues and feedback requests

OPEN (1)

Confirmar contraseña

Opened 5 days ago by mfarías · open

GC

RECENT (6)

Todo esto se refiere al blanco

Blanco

Posted just now by cpinkus

@cerati:

Posted 4 minutes ago by cpinkus

@cerati Por qué 600?

Posted yesterday by mfarías

1

PE-4 Agregar ciudad al formulario de registro

Opened yesterday by mfarías · approved

GC

6

Esto funciona en todos los browsers?

Posted 5 days ago by mfarías

1

Does it work in all browsers?

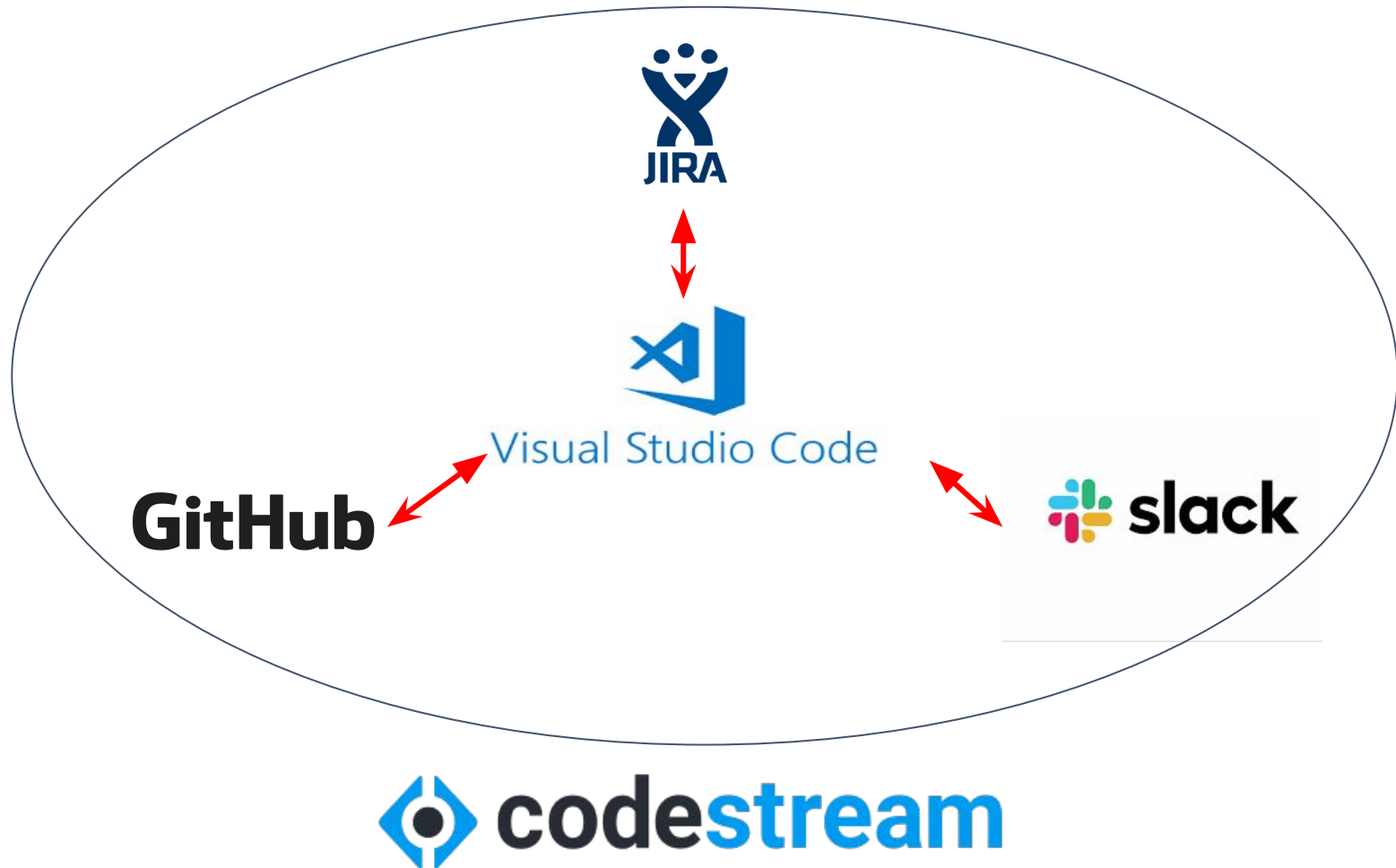
Posted 5 days ago by mfarías

Pro Tip!

Created before yesterday: created:<yesterday



# Resumen del curso



# Implementación del flujo moderno

## Uso/Efecto



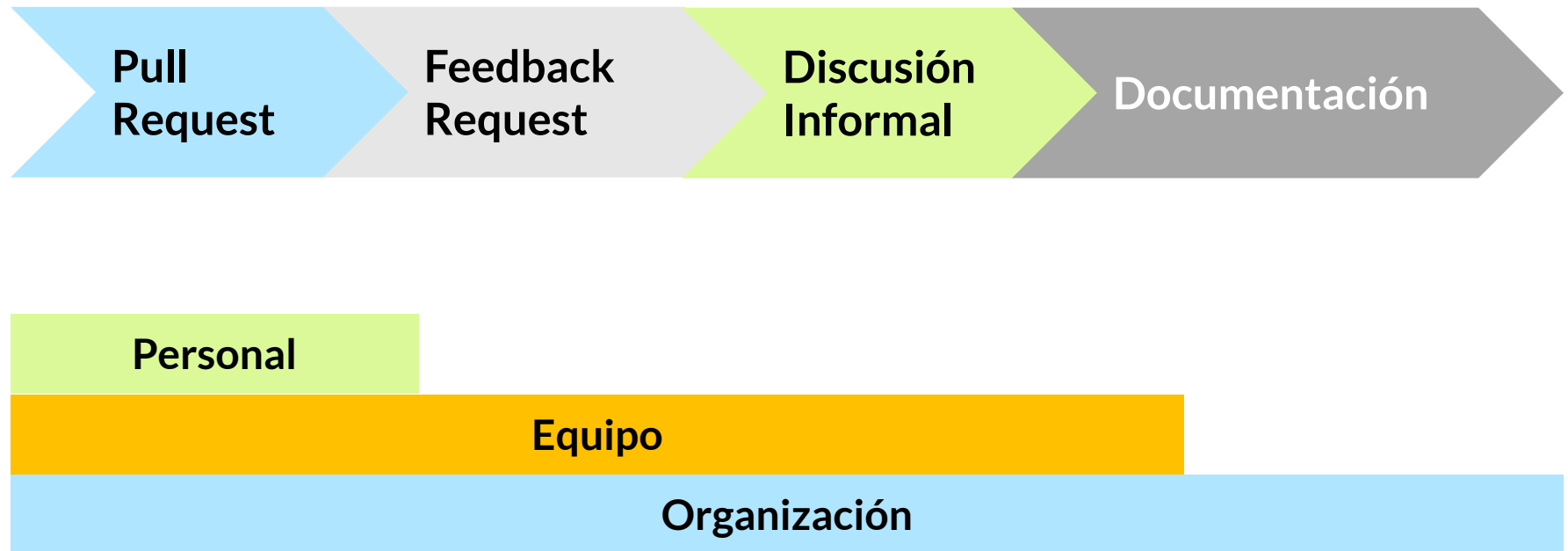
- Pull Request
- Issue tracker
- Comentarios

- Pull Request
- Issue tracker
- Comentarios
- Feedback Request
- Code Chat
- Transparencia

- Pull Request
- Issue tracker
- Comentarios
- Feedback Request
- Code Chat
- Transparencia
- Documentación
- Administración
- Análisis

# Evolución del flujo

## El camino hacia *Shift Left*



# Ahorro de pasos: GitHub, Jira, Slack

Sin integración

59

Pasos

Con integración

14

Pasos

# Documentación automática



# Traducción del Proyecto Open Source CodeStream



---

# 6 predicciones sobre el futuro del desarrollo de software



# El futuro del desarrollo de software

1. El cambio a desarrollo remoto será permanente.





# El futuro del desarrollo de software

2. Git es el presente y el futuro.



# El futuro del desarrollo de software

3. Los environments vendrán precargados con los repositorios.



# El futuro del desarrollo de software

4. Los environments y los lenguajes de programación se volverán más especializados.

# El futuro del desarrollo de software

5. Las barreras entre el desarrollo local, el desarrollo cloud, el desarrollo en pares y el desarrollo en equipo, irán desapareciendo.



# El futuro del desarrollo de software

6. El desarrollo de software será más colaborativo y más transparente. Al ser tu copiloto.



# Comentarios

- ¡Cuéntame qué te pareció el curso!
- Siguenos en Twitter @teamcodestream.
- Si te resultó útil el curso, implementa la tecnología y el Flujo de Desarrollo Moderno.
- Presenta el examen.
- Comparte el diploma con tus redes y colegas.