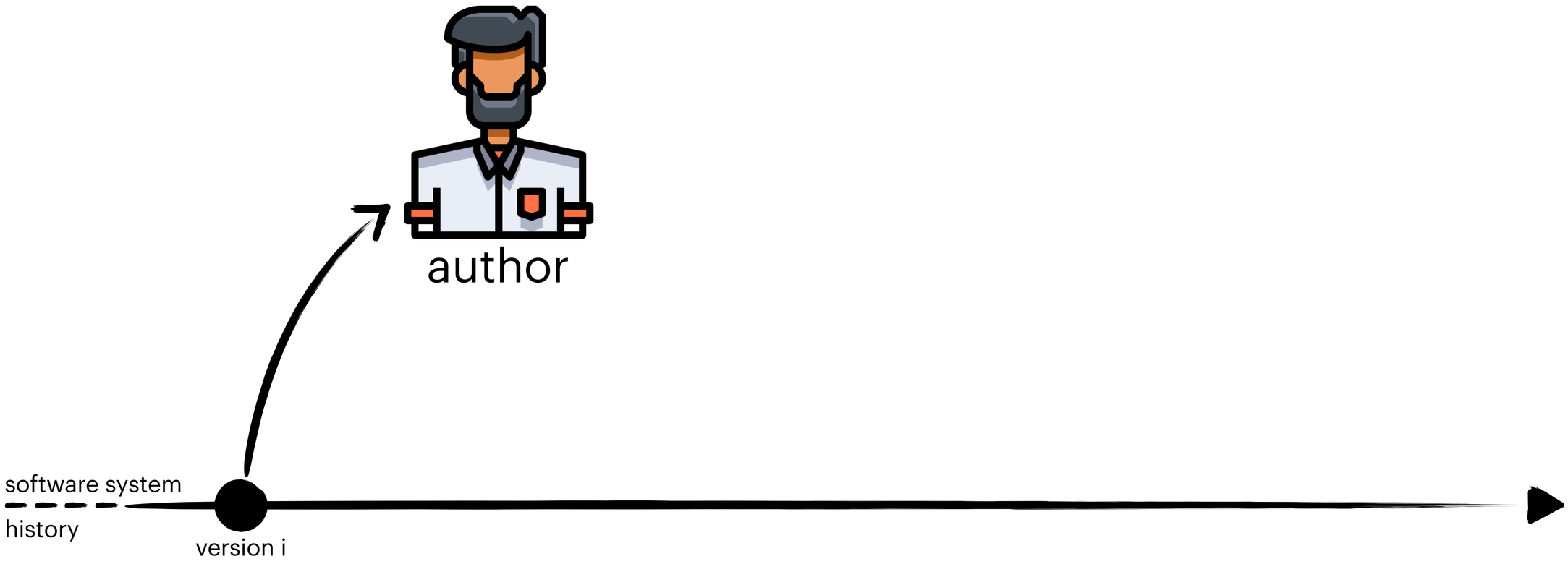
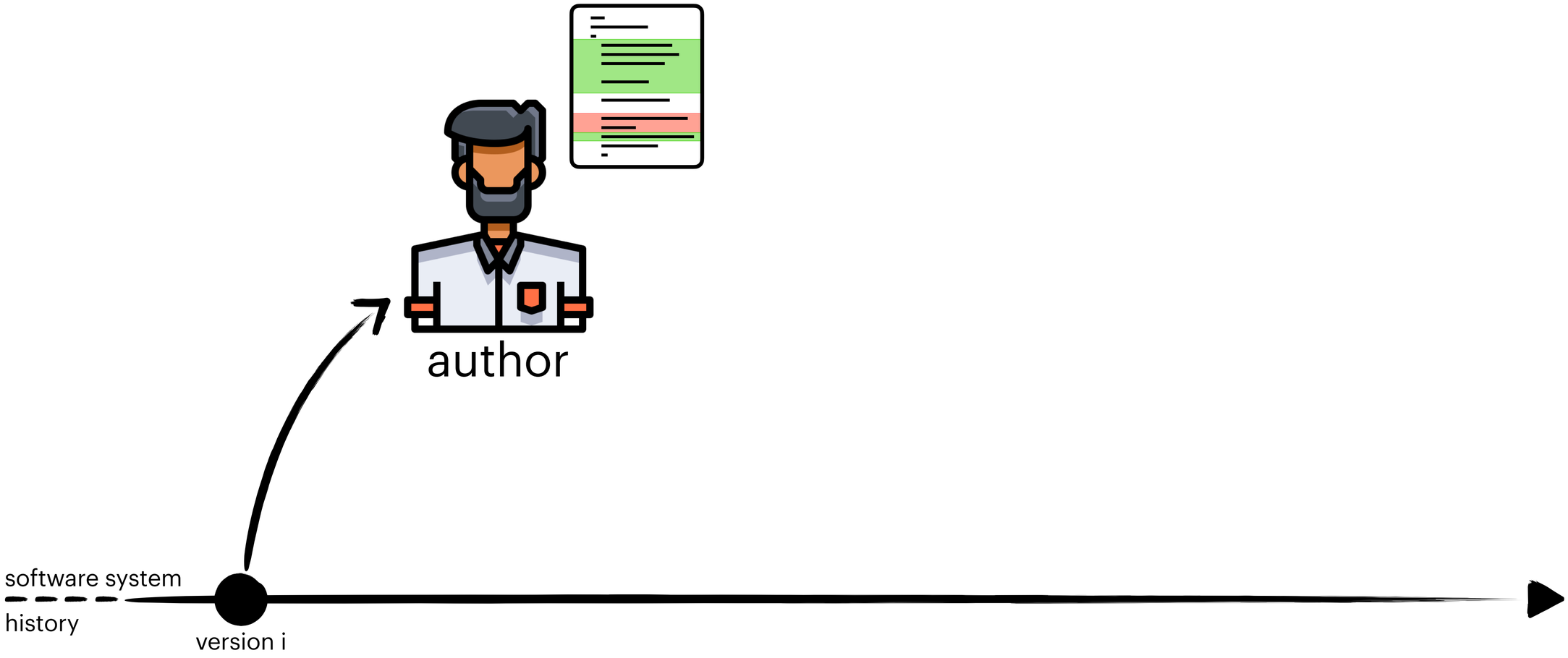
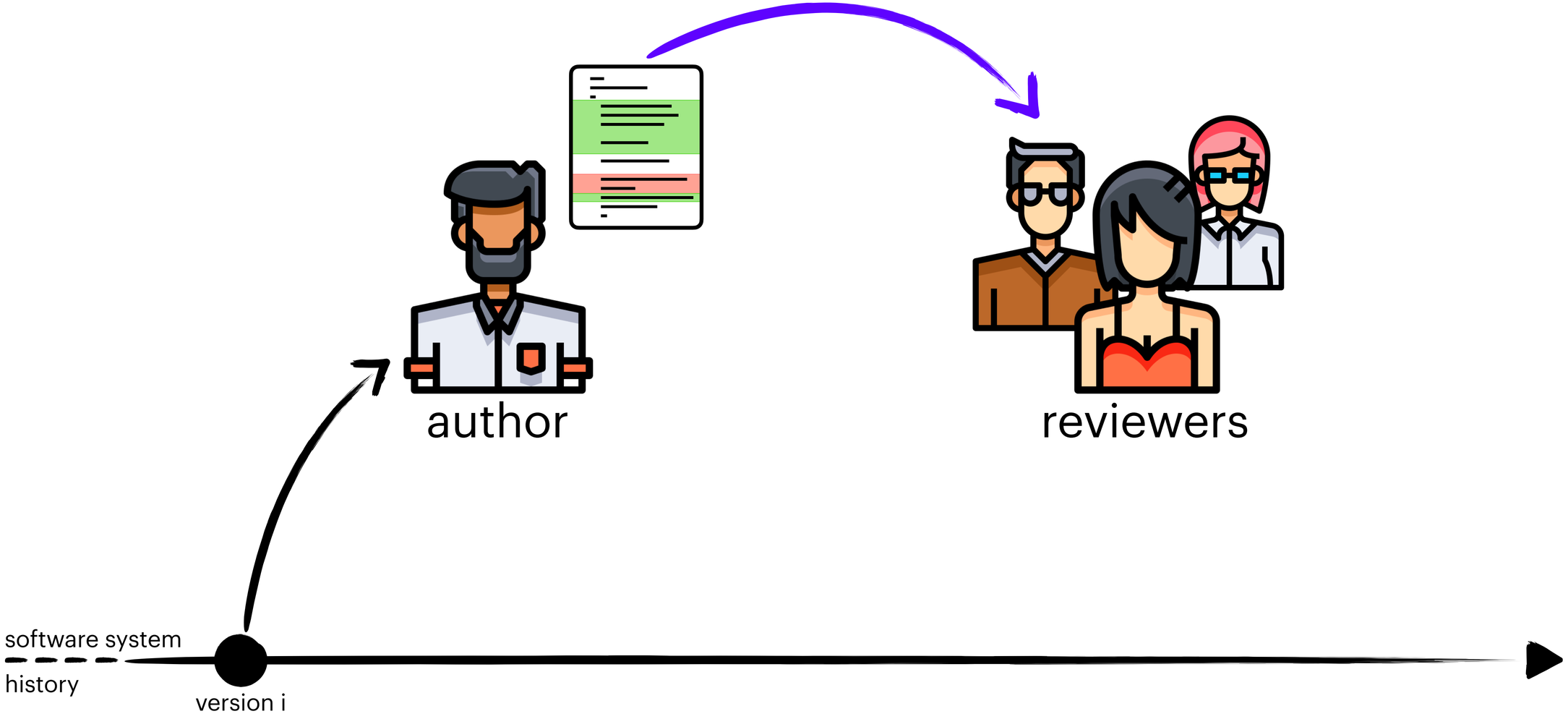
Code review effectiveness

How code review works (and doesn't) in the real world.

Alberto Bacchelli
zest
University of Zurich[UZH]

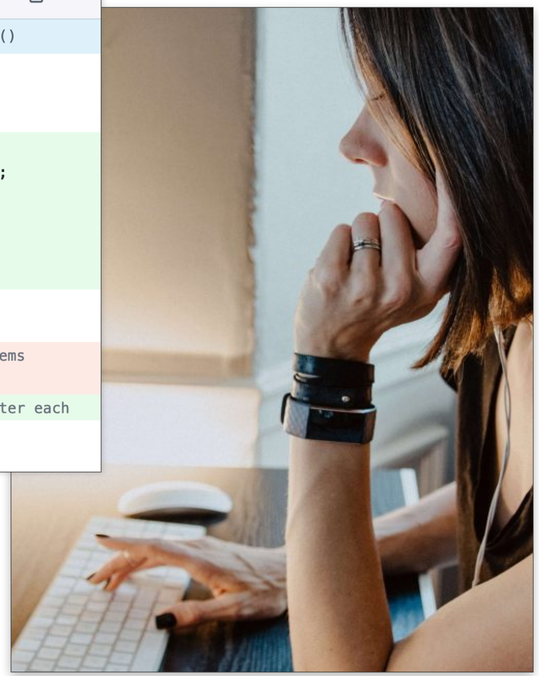

author
software system
history
version i

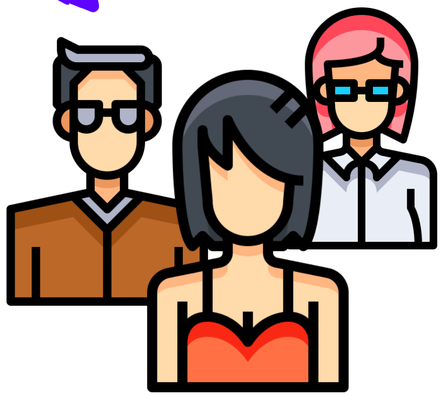

author
software system
history
version i


author
reviewers
software system
history
version i

33 src/System.Collections.Immutable/tests/ImmutableListTest.cs

```diff
@@ -164,29 +164,34 @@ public void AddRangeOptimizationsTest()
 164 164         [Fact]
 165 165         public void AddRangeBalanceTest()
 166 166         {
     167 +           int randSeed = (int)DateTime.Now.Ticks;
     168 +           Console.WriteLine("Random seed: {0}", randSeed);
     169 +           var random = new Random(randSeed);
     170 +
     171 +           int expectedTotalSize = 0;
     172 +
 167 173             var list = ImmutableList<int>.Empty;
 168 174
 169     -           // Add batches of 32, 128 times, giving 4096 items
 170     -           int batchSize = 32;
     175 +           // Add some small batches, verifying balance after each
 171 176             for (int i = 0; i < 128; i++)
 172 177             {
```





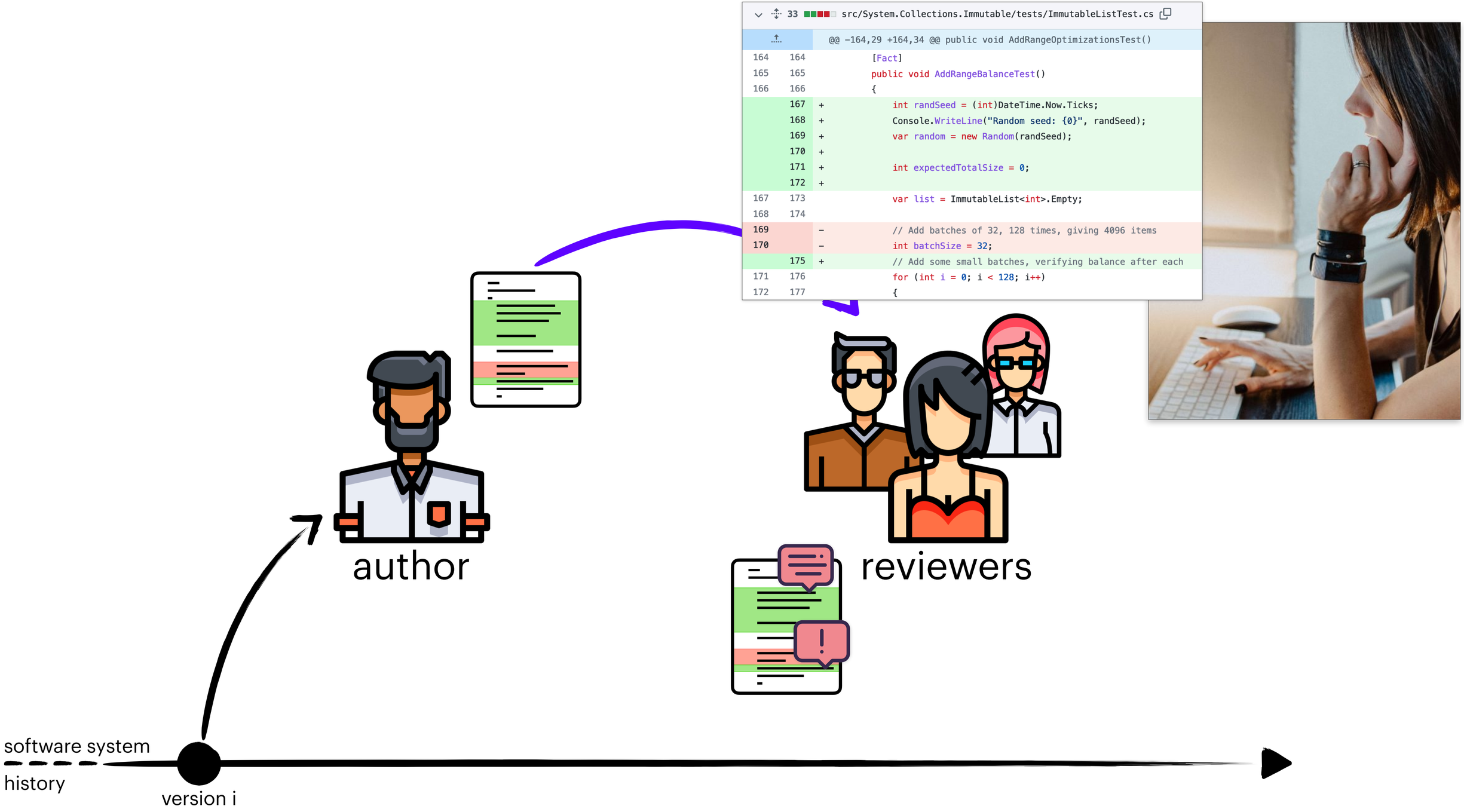
author

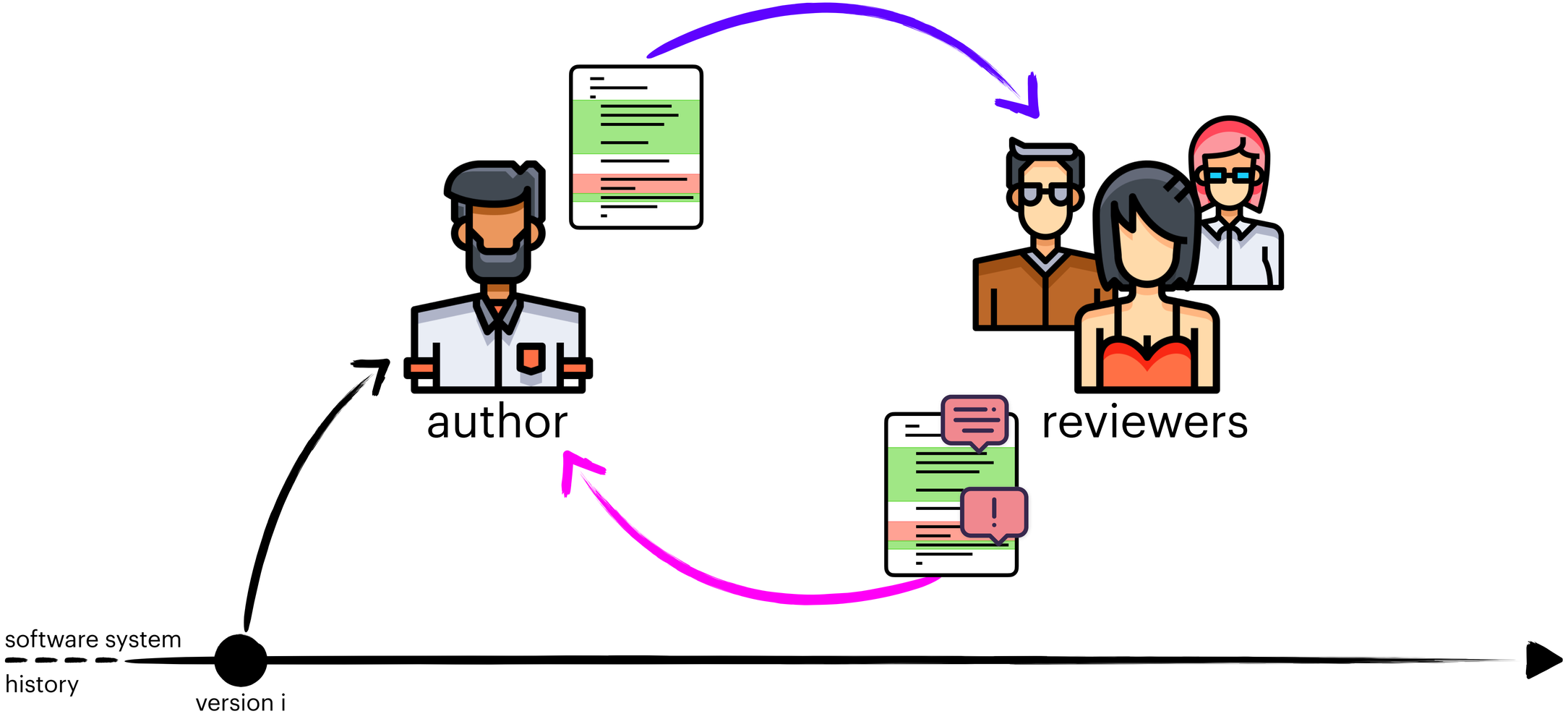
reviewers

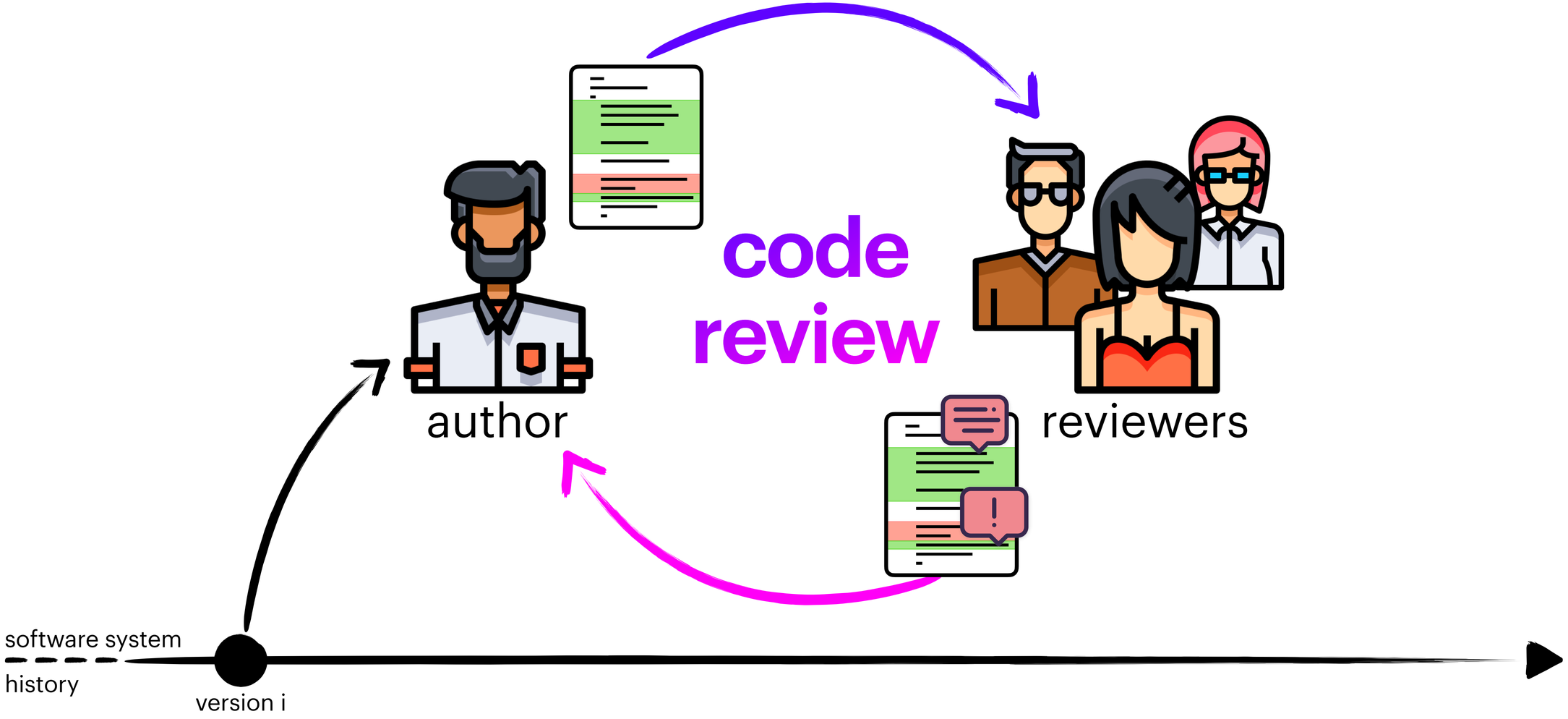
software system

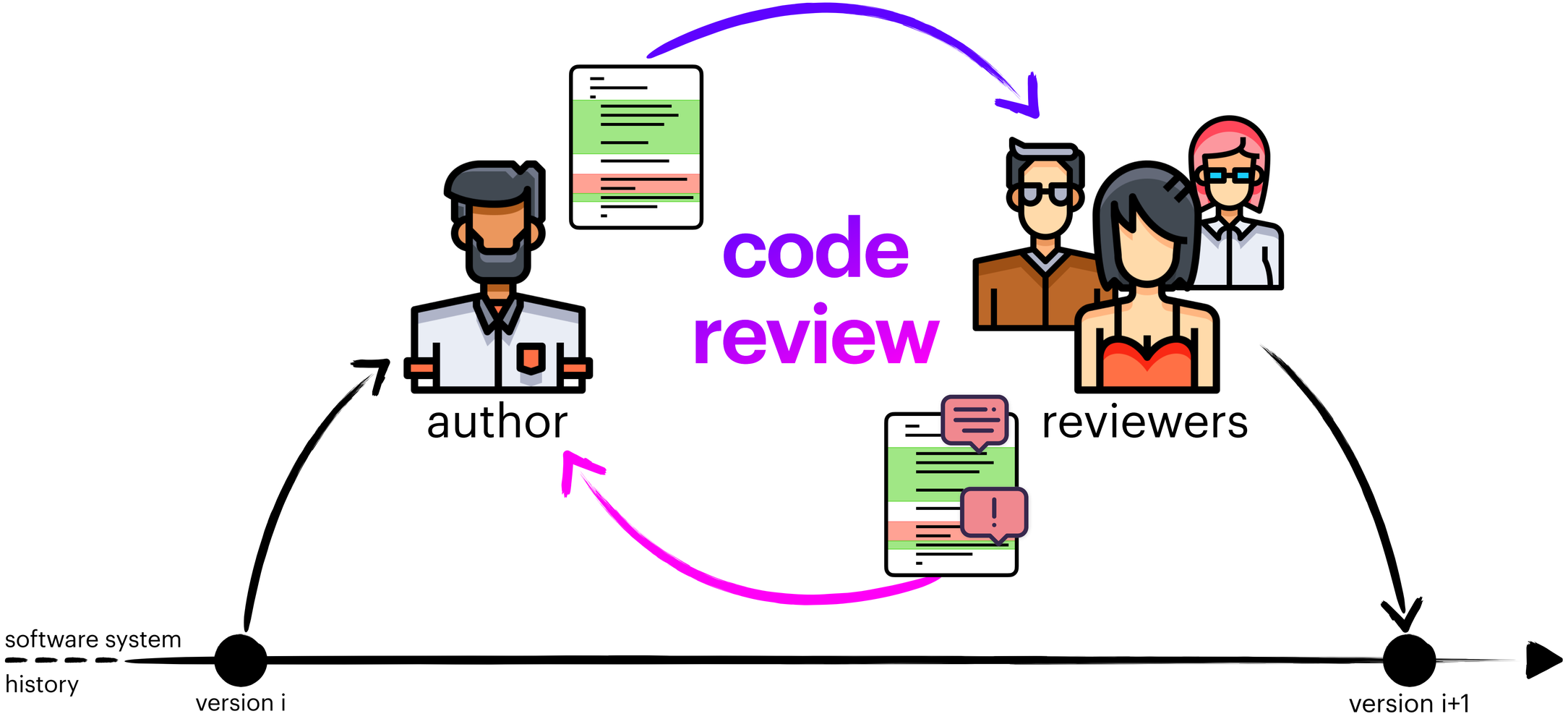
history

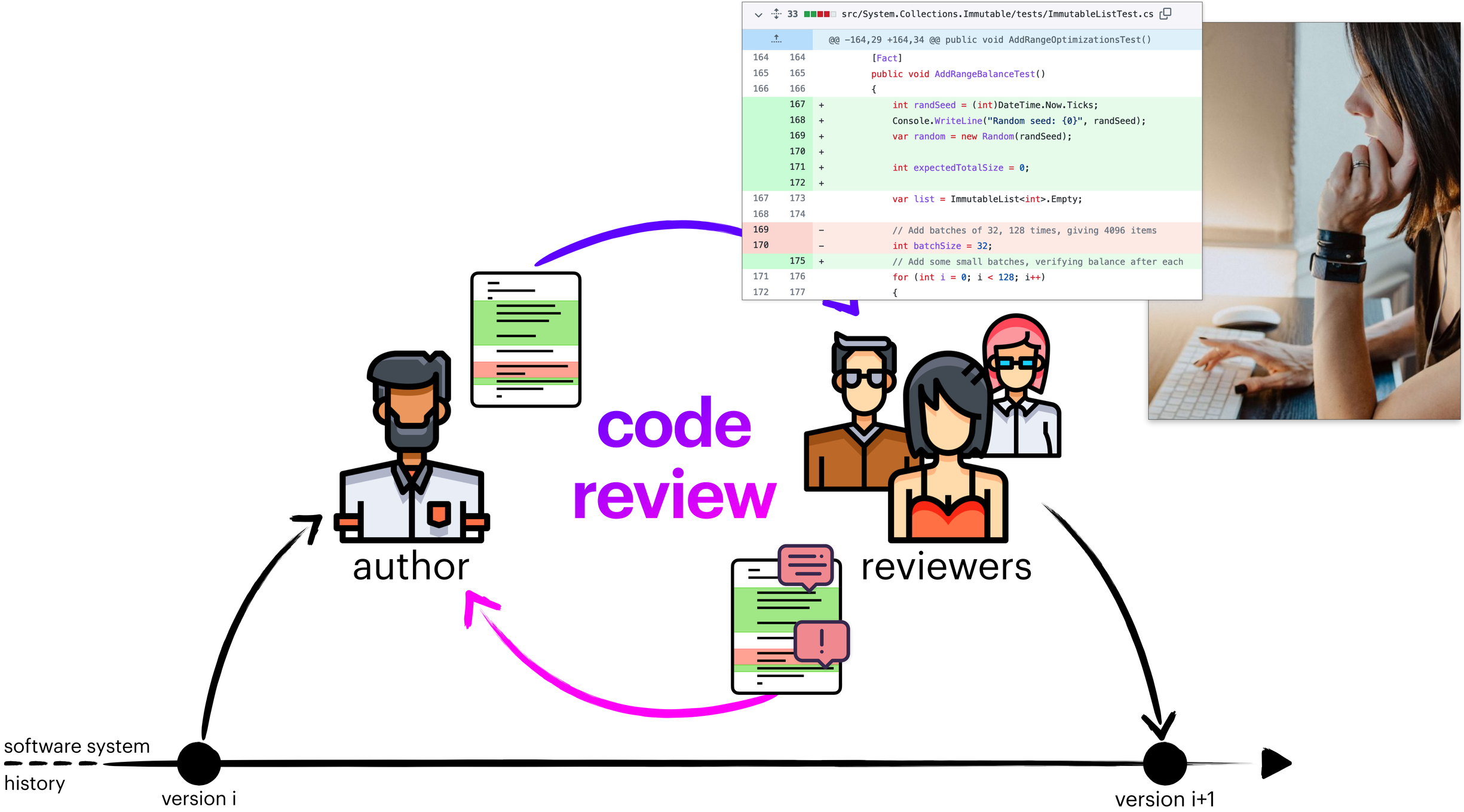
version i


33 src/System.Collections.Immutable/tests/ImmutableListTest.cs
@@ -164,29 +164,34 @@ public void AddRangeOptimizationsTest()
164 164 [Fact]
165 165 public void AddRangeBalanceTest()
166 166 {
+ 167 int randSeed = (int)DateTime.Now.Ticks;
+ 168 Console.WriteLine("Random seed: {0}", randSeed);
+ 169 var random = new Random(randSeed);
+ 170
+ 171 int expectedTotalSize = 0;
+ 172
167 173 var list = ImmutableList<int>.Empty;
168 174
169 - // Add batches of 32, 128 times, giving 4096 items
170 - int batchSize = 32;
+ 175 // Add some small batches, verifying balance after each
171 176 for (int i = 0; i < 128; i++)
172 177 {
author
reviewers
software system
history
version i


author
reviewers
software system
history
version i


code
review
author
reviewers
software system
history
version i


code
review
author
reviewers
software system
history
version i
version i+1

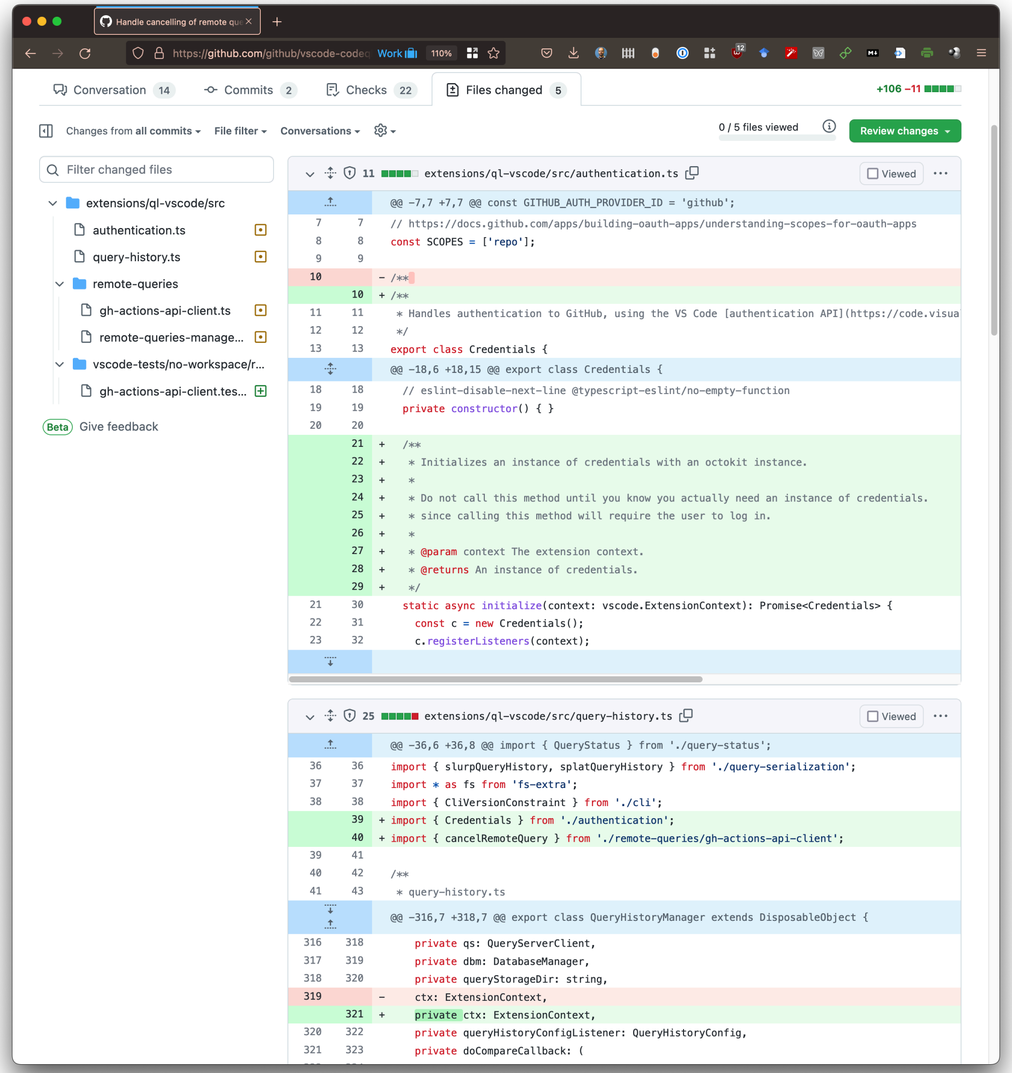

33 src/System.Collections.Immutable/tests/ImmutableListTest.cs
@@ -164,29 +164,34 @@ public void AddRangeOptimizationsTest()
164 164 [Fact]
165 165 public void AddRangeBalanceTest()
166 166 {
167 + int randSeed = (int)DateTime.Now.Ticks;
168 + Console.WriteLine("Random seed: {0}", randSeed);
169 + var random = new Random(randSeed);
170 +
171 + int expectedTotalSize = 0;
172 +
167 173 var list = ImmutableList<int>.Empty;
168 174
169 – // Add batches of 32, 128 times, giving 4096 items
170 – int batchSize = 32;
175 + // Add some small batches, verifying balance after each
171 176 for (int i = 0; i < 128; i++)
172 177 {
code review
author
reviewers
software system
history
version i
version i+1

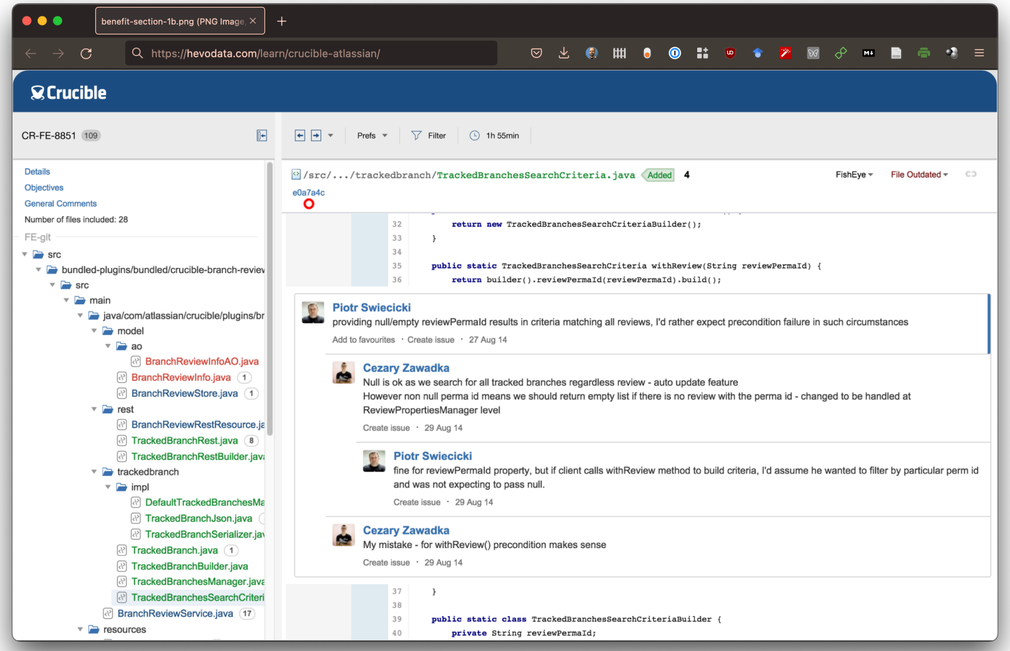
code review
some facts

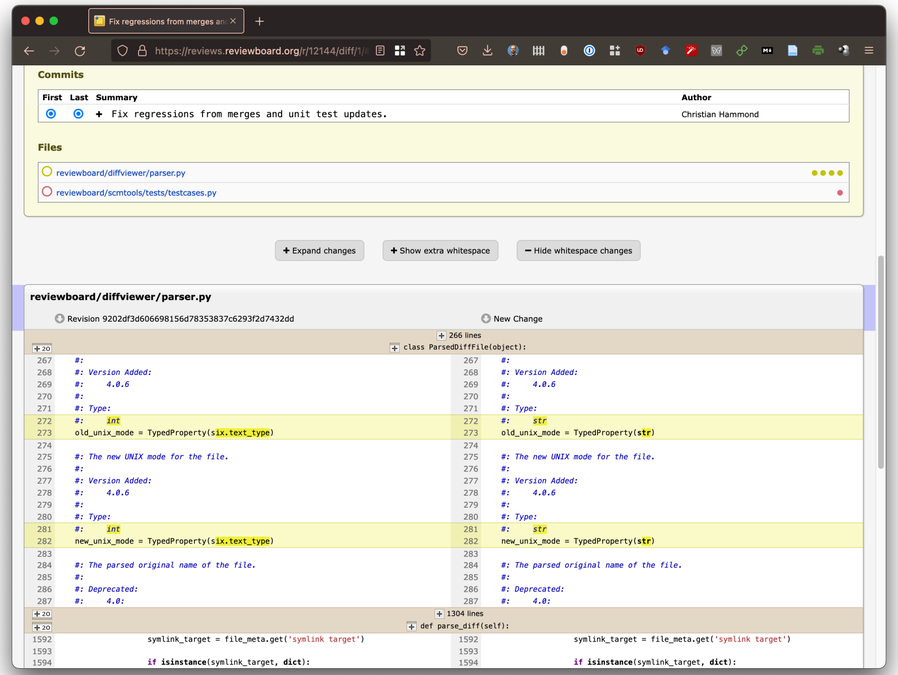
- About 70% of developers spend 2 to 8 hours a week reviewing code.
  [Stack Overflow Dev Survey 2019]
- In 2021, 170M pull requests have been merged in GitHub.
  [The 2021 State of the Octoverse]
- Most code changes at Microsoft and Google are reviewed.

code review

tools







code review

Handle cancelling of remote que

https://github.com/github/vscode-codeq

Conversation 14 | Commits 2 | Checks 22 | Files changed 5

Changes from all commits | File filter | Conversations | 0 / 5 files

Filter changed files

- extensions/ql-vscode/src
  - authentication.ts
  - query-history.ts
  - remote-queries
    - gh-actions-api-client.ts
    - remote-queries-manage...
  - vscode-tests/no-workspace/r...
    - gh-actions-api-client.tes...

Beta Give feedback

11 extensions/ql-vscode/src/authentication.ts

```
@@ -7,7 +7,7 @@ const GITHUB_AUTH_PROVIDER_ID = 'github';
 7  7  // https://docs.github.com/apps/building-oauth-apps/understand
 8  8  const SCOPES = ['repo'];
 9  9
10     - /**
   10  + /**
11 11   * Handles authentication to GitHub, using the VS Code [authen
12 12   */
13 13  export class Credentials {
@@ -18,6 +18,15 @@ export class Credentials {
18 18    // eslint-disable-next-line @typescript-eslint/no-empty-func
19 19    private constructor() { }
20 20
   21  +  /**
   22  +   * Initializes an instance of credentials with an octokit in
   23  +   *
   24  +   * Do not call this method until you know you actually need
   25  +   * since calling this method will require the user to log in
   26  +   *
   27  +   * @param context The extension context.
   28  +   * @returns An instance of credentials.
   29  +   */
21 30    static async initialize(context: vscode.ExtensionContext): P
22 31      const c = new Credentials();
23 32      c.registerListeners(context);
```

25 extensions/ql-vscode/src/query-history.ts

```
@@ -36,6 +36,8 @@ import { QueryStatus } from './query-status'
36 36  import { slurpQueryHistory, splatQueryHistory } from './query-
37 37  import * as fs from 'fs-extra';
38 38  import { CliVersionConstraint } from './cli';
   39  + import { Credentials } from './authentication';
   40  + import { cancelRemoteQuery } from './remote-queries/gh-actions
```

code review

Handle cancelling of remote que

https://github.com/github/vscode-codeq

Conversation 14 | Commits 2 | Checks 22 | Files changed 5

Changes from all commits | File filter | Conversations | 0 / 5 files

Filter changed files

- extensions/ql-vscode/src
  - authentication.ts
  - query-history.ts
  - remote-queries
    - gh-actions-api-client.ts
    - remote-queries-manage...
  - vscode-tests/no-workspace/r...
    - gh-actions-api-client.tes...

Beta Give feedback

11 extensions/ql-vscode/src/authentication.ts

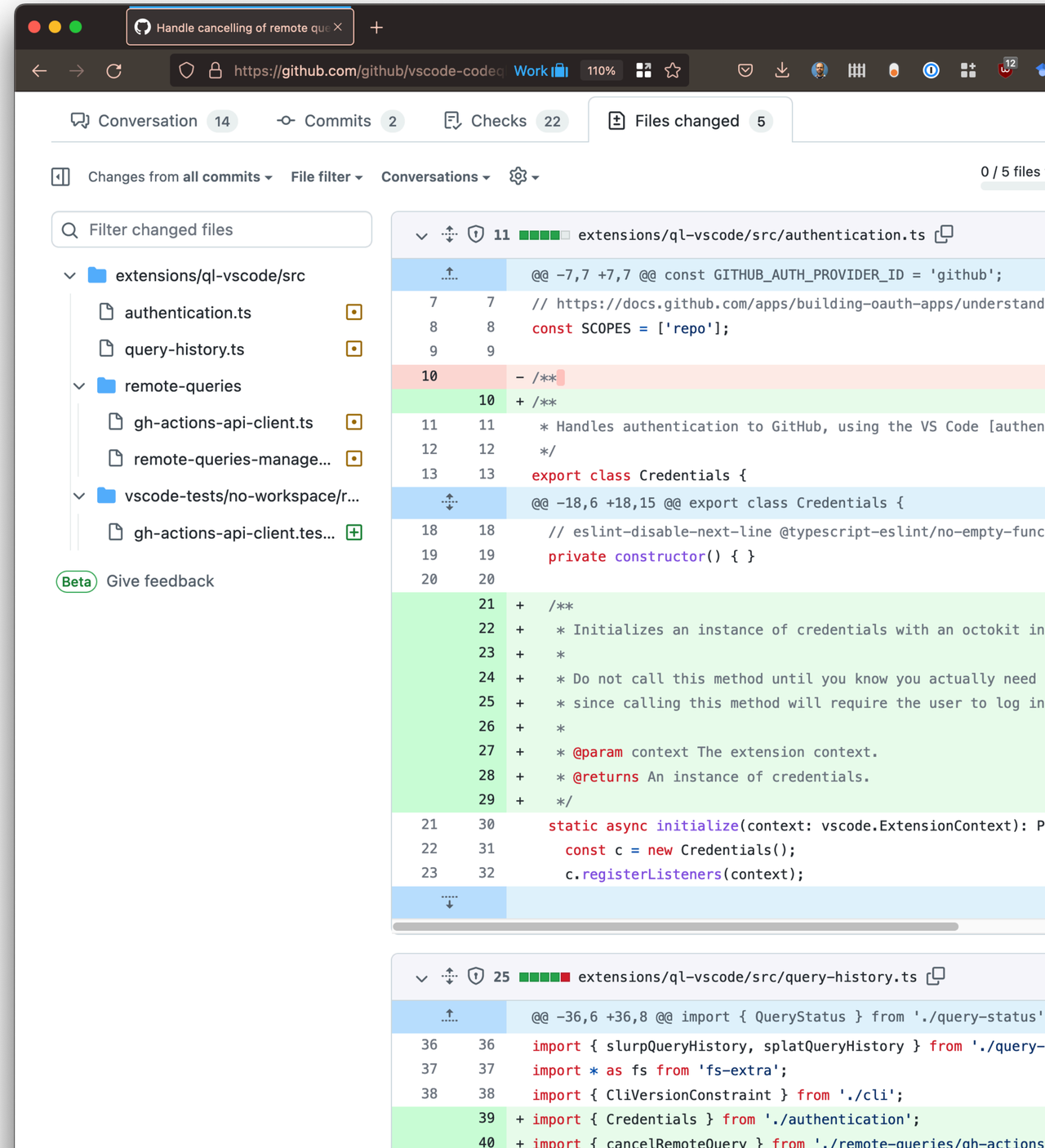
```
@@ -7,7 +7,7 @@ const GITHUB_AUTH_PROVIDER_ID = 'github';
 7  7  // https://docs.github.com/apps/building-oauth-apps/understand
 8  8  const SCOPES = ['repo'];
 9  9
10     - /**
   10  + /**
11 11   * Handles authentication to GitHub, using the VS Code [authen
12 12   */
13 13  export class Credentials {
@@ -18,6 +18,15 @@ export class Credentials {
18 18    // eslint-disable-next-line @typescript-eslint/no-empty-func
19 19    private constructor() { }
20 20
   21  +  /**
   22  +   * Initializes an instance of credentials with an octokit in
   23  +   *
   24  +   * Do not call this method until you know you actually need
   25  +   * since calling this method will require the user to log in
   26  +   *
   27  +   * @param context The extension context.
   28  +   * @returns An instance of credentials.
   29  +   */
21 30    static async initialize(context: vscode.ExtensionContext): P
22 31      const c = new Credentials();
23 32      c.registerListeners(context);
```

25 extensions/ql-vscode/src/query-history.ts

```
@@ -36,6 +36,8 @@ import { QueryStatus } from './query-status'
36 36  import { slurpQueryHistory, splatQueryHistory } from './query-
37 37  import * as fs from 'fs-extra';
38 38  import { CliVersionConstraint } from './cli';
   39  + import { Credentials } from './authentication';
   40  + import { cancelRemoteQuery } from './remote-queries/gh-actions
```

code review

reviewers' behavior

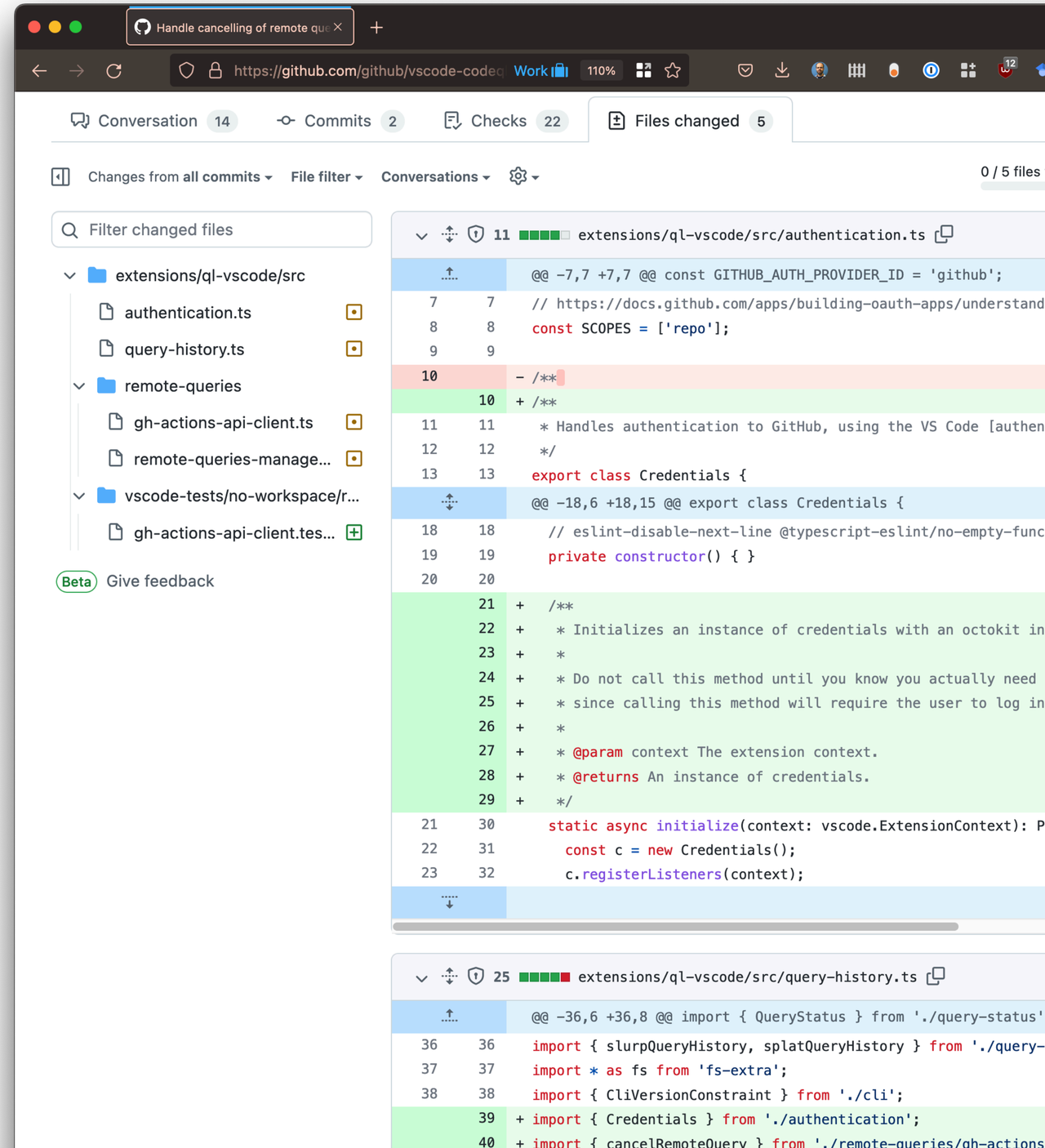


code review

reviewers' behavior

- In more than 50% of the reviews, the reviewer started with the file presented first.
- In almost 40% of the navigations, the reviewer went to the next file in order.

[Baum, Schneider, Bacchelli - ICSME 2017]



code review

test files

Handle cancelling of remote que

https://github.com/github/vscode-codeq Work 110%

Conversation 14 | Commits 2 | Checks 22 | Files changed 5

Changes from all commits | File filter | Conversations | 0 / 5 files

Filter changed files

- extensions/ql-vscode/src
  - authentication.ts
  - query-history.ts
  - remote-queries
    - gh-actions-api-client.ts
    - remote-queries-manage...
- vscode-tests/no-workspace/r...
  - gh-actions-api-client.tes...

Beta Give feedback

11 extensions/ql-vscode/src/authentication.ts

```
@@ -7,7 +7,7 @@ const GITHUB_AUTH_PROVIDER_ID = 'github';
7  7   // https://docs.github.com/apps/building-oauth-apps/understand
8  8   const SCOPES = ['repo'];
9  9
10     - /**
   10  + /**
11 11   * Handles authentication to GitHub, using the VS Code [authen
12 12   */
13 13  export class Credentials {
@@ -18,6 +18,15 @@ export class Credentials {
18 18    // eslint-disable-next-line @typescript-eslint/no-empty-func
19 19    private constructor() { }
20 20
   21  +  /**
   22  +   * Initializes an instance of credentials with an octokit in
   23  +   *
   24  +   * Do not call this method until you know you actually need
   25  +   * since calling this method will require the user to log in
   26  +   *
   27  +   * @param context The extension context.
   28  +   * @returns An instance of credentials.
   29  +   */
21 30    static async initialize(context: vscode.ExtensionContext): P
22 31      const c = new Credentials();
23 32      c.registerListeners(context);
```

25 extensions/ql-vscode/src/query-history.ts

```
@@ -36,6 +36,8 @@ import { QueryStatus } from './query-status'
36 36  import { slurpQueryHistory, splatQueryHistory } from './query-
37 37  import * as fs from 'fs-extra';
38 38  import { CliVersionConstraint } from './cli';
   39  + import { Credentials } from './authentication';
   40  + import { cancelRemoteQuery } from './remote-queries/gh-actions
```

code review

test files

- Developers see test code as less important than production code.

Handle cancelling of remote que

https://github.com/github/vscode-codeq Work 110%

Conversation 14 | Commits 2 | Checks 22 | Files changed 5

Changes from all commits | File filter | Conversations | 0 / 5 files

Filter changed files

- extensions/ql-vscode/src
  - authentication.ts
  - query-history.ts
  - remote-queries
    - gh-actions-api-client.ts
    - remote-queries-manage...
- vscode-tests/no-workspace/r...
  - gh-actions-api-client.tes...

Beta Give feedback

11 extensions/ql-vscode/src/authentication.ts

```
@@ -7,7 +7,7 @@ const GITHUB_AUTH_PROVIDER_ID = 'github';
7  7   // https://docs.github.com/apps/building-oauth-apps/understand
8  8   const SCOPES = ['repo'];
9  9
10     - /**
   10  + /**
11 11   * Handles authentication to GitHub, using the VS Code [authen
12 12   */
13 13  export class Credentials {
@@ -18,6 +18,15 @@ export class Credentials {
18 18    // eslint-disable-next-line @typescript-eslint/no-empty-func
19 19    private constructor() { }
20 20
   21  +  /**
   22  +   * Initializes an instance of credentials with an octokit in
   23  +   *
   24  +   * Do not call this method until you know you actually need
   25  +   * since calling this method will require the user to log in
   26  +   *
   27  +   * @param context The extension context.
   28  +   * @returns An instance of credentials.
   29  +   */
21 30    static async initialize(context: vscode.ExtensionContext): P
22 31      const c = new Credentials();
23 32      c.registerListeners(context);
```

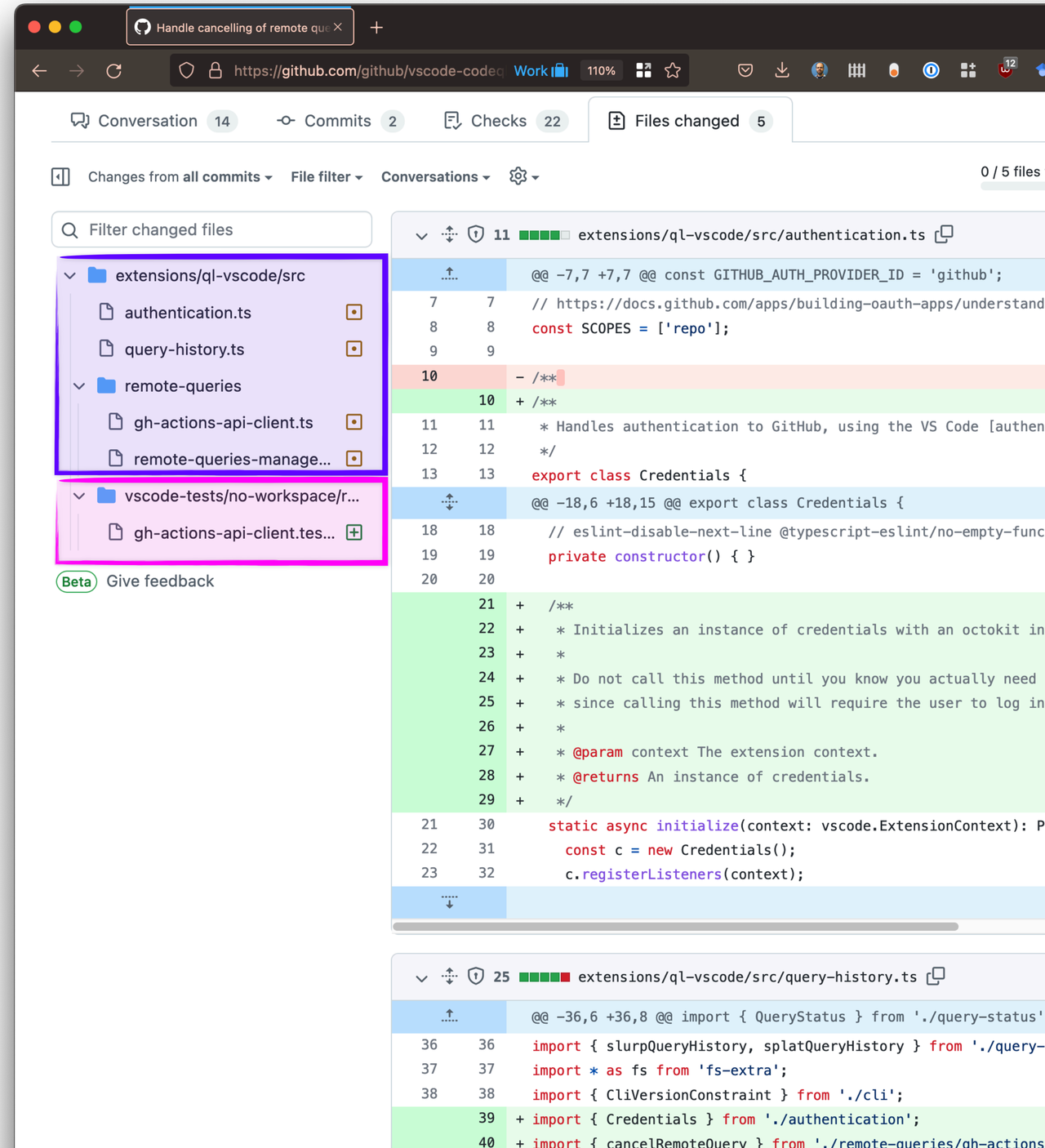
25 extensions/ql-vscode/src/query-history.ts

```
@@ -36,6 +36,8 @@ import { QueryStatus } from './query-status'
36 36  import { slurpQueryHistory, splatQueryHistory } from './query-
37 37  import * as fs from 'fs-extra';
38 38  import { CliVersionConstraint } from './cli';
   39  + import { Credentials } from './authentication';
   40  + import { cancelRemoteQuery } from './remote-queries/gh-actions
```

code review

test files

- Developers see test code as less important than production code.
- Test files are almost twice less likely to be discussed during code review when together with production files.

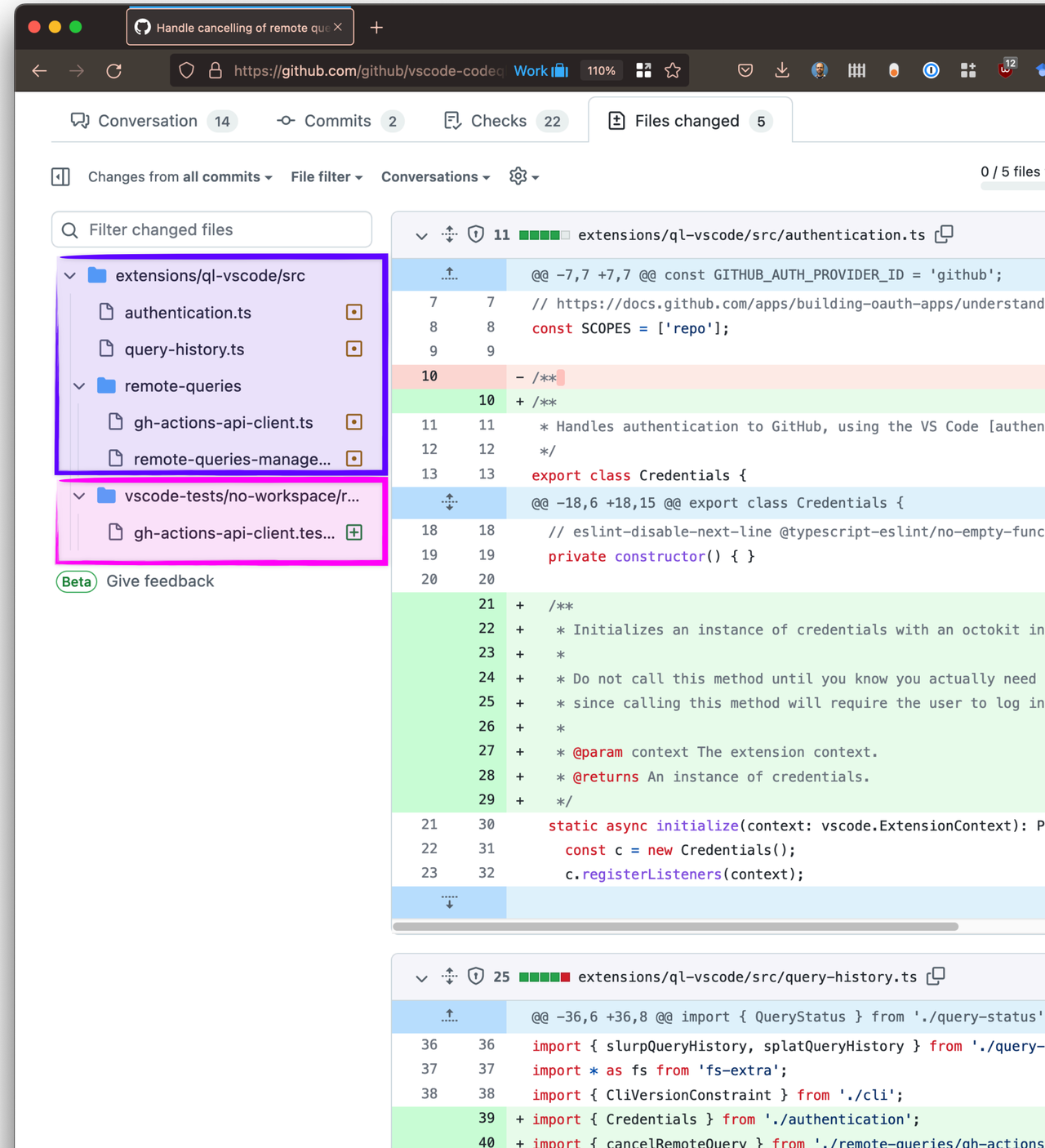


code review

test files

- Developers see test code as less important than production code.
- Test files are almost twice less likely to be discussed during code review when together with production files.

[Spadini, Aniche, Storey, Bruntink, Bacchelli - ICSE 2018]

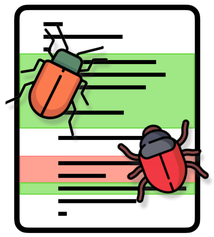


code review

test order experiment

code review

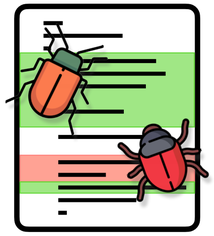
test order experiment



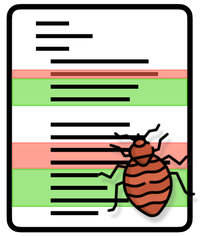
production.java

code review

test order experiment



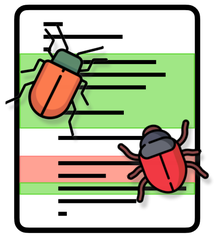
production.java



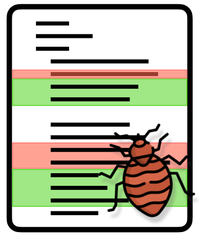
test.java

code review

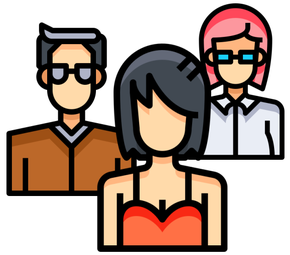
test order experiment



production.java



test.java

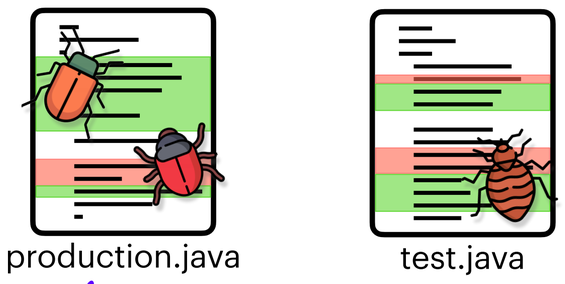


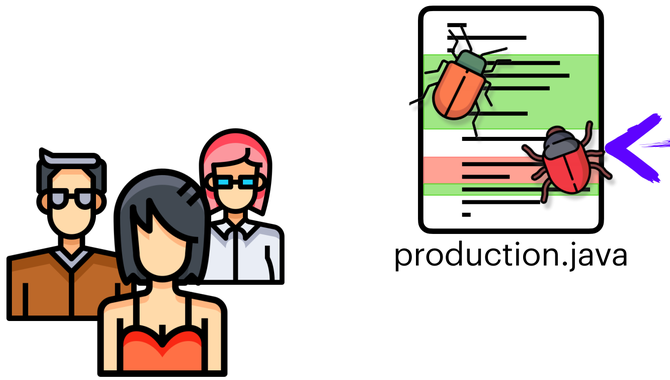
participants

participants

code review

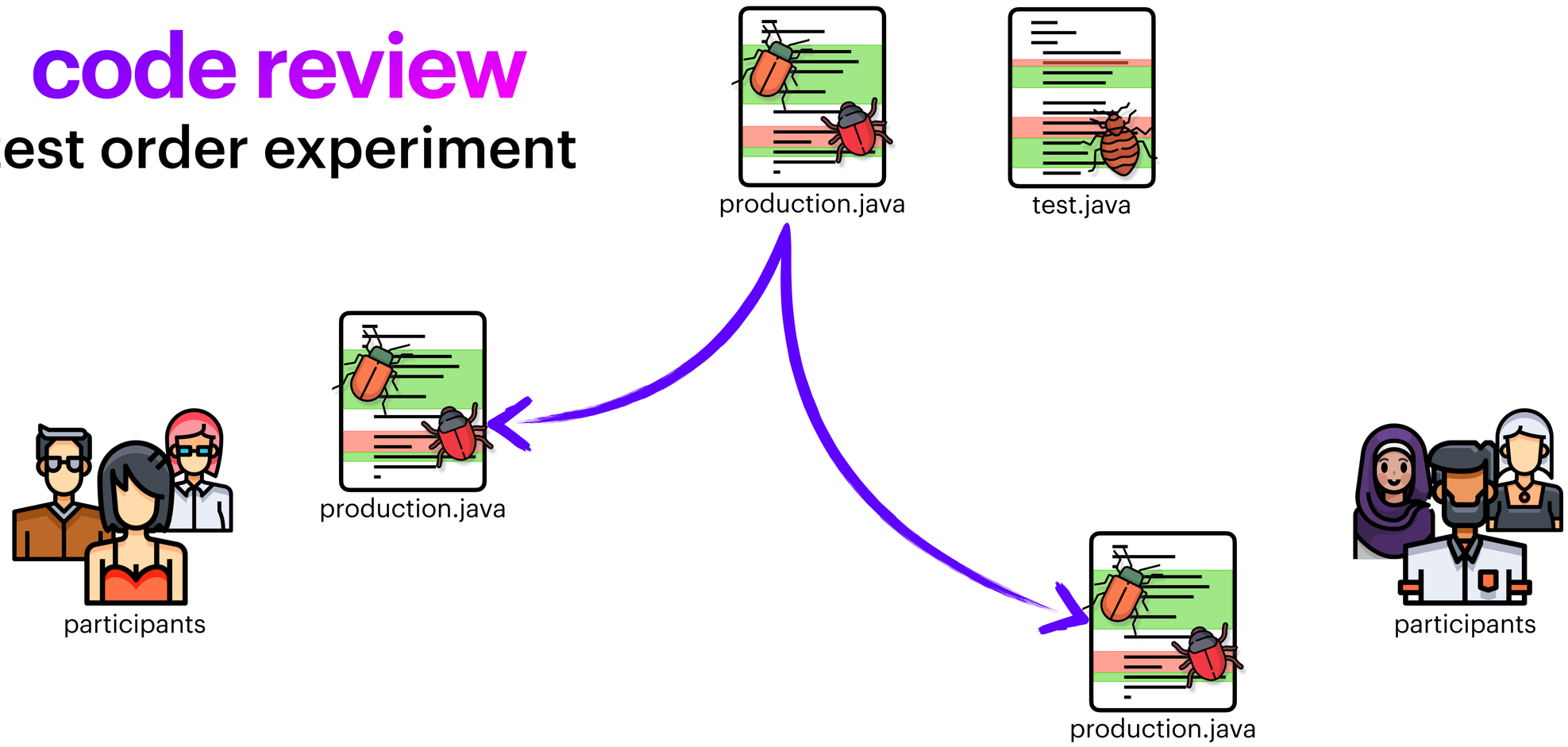
test order experiment





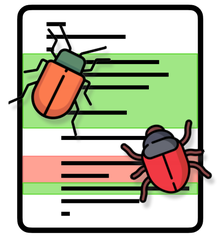
code review

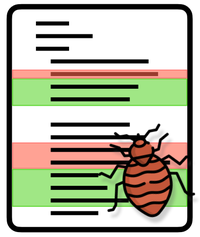
test order experiment

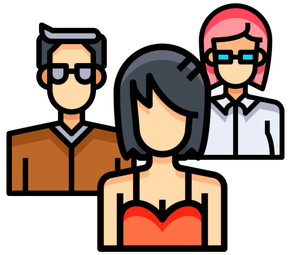


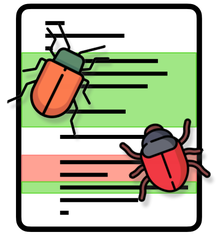
code review

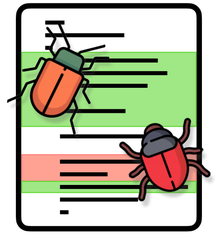
test order experiment


production.java


test.java


participants

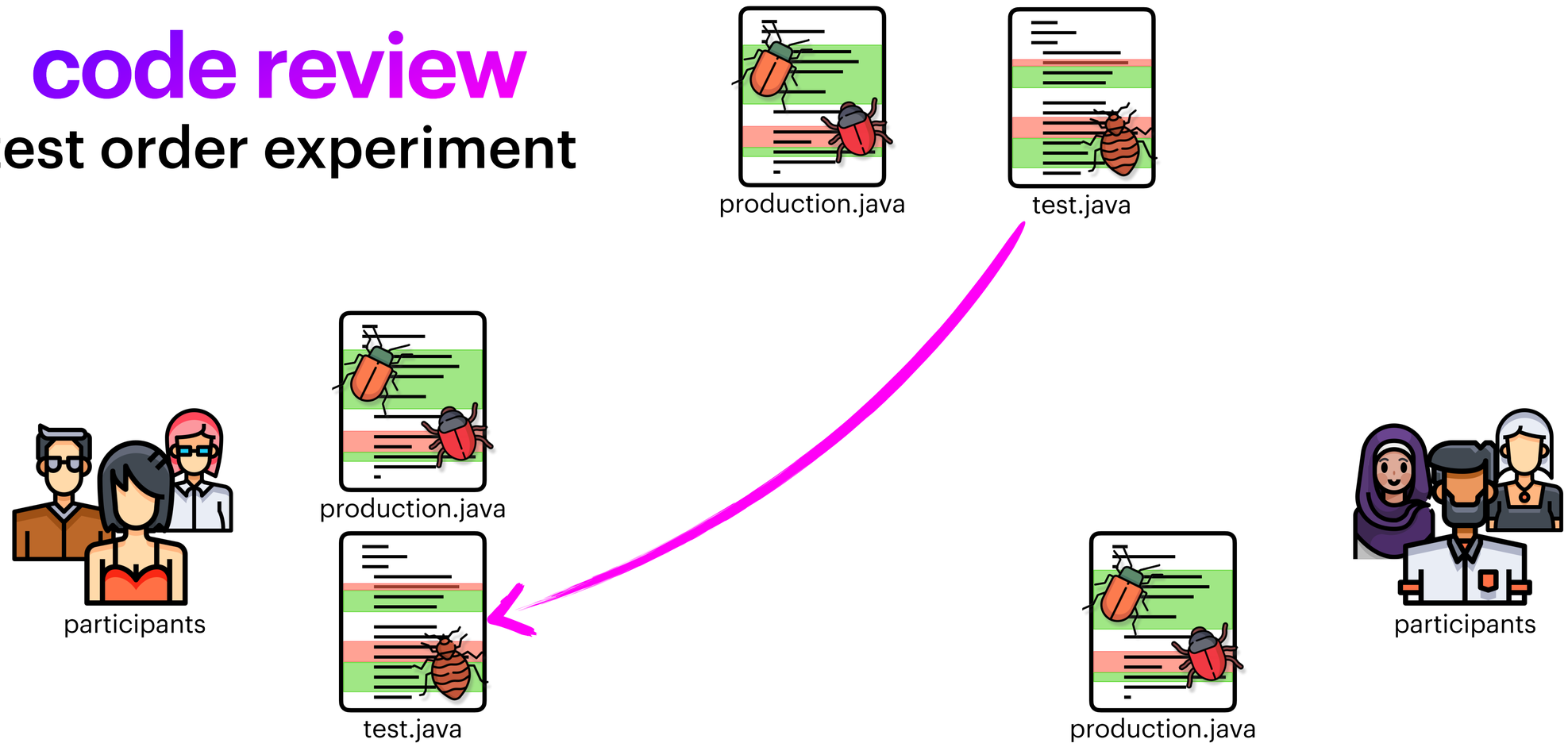

production.java


production.java


participants

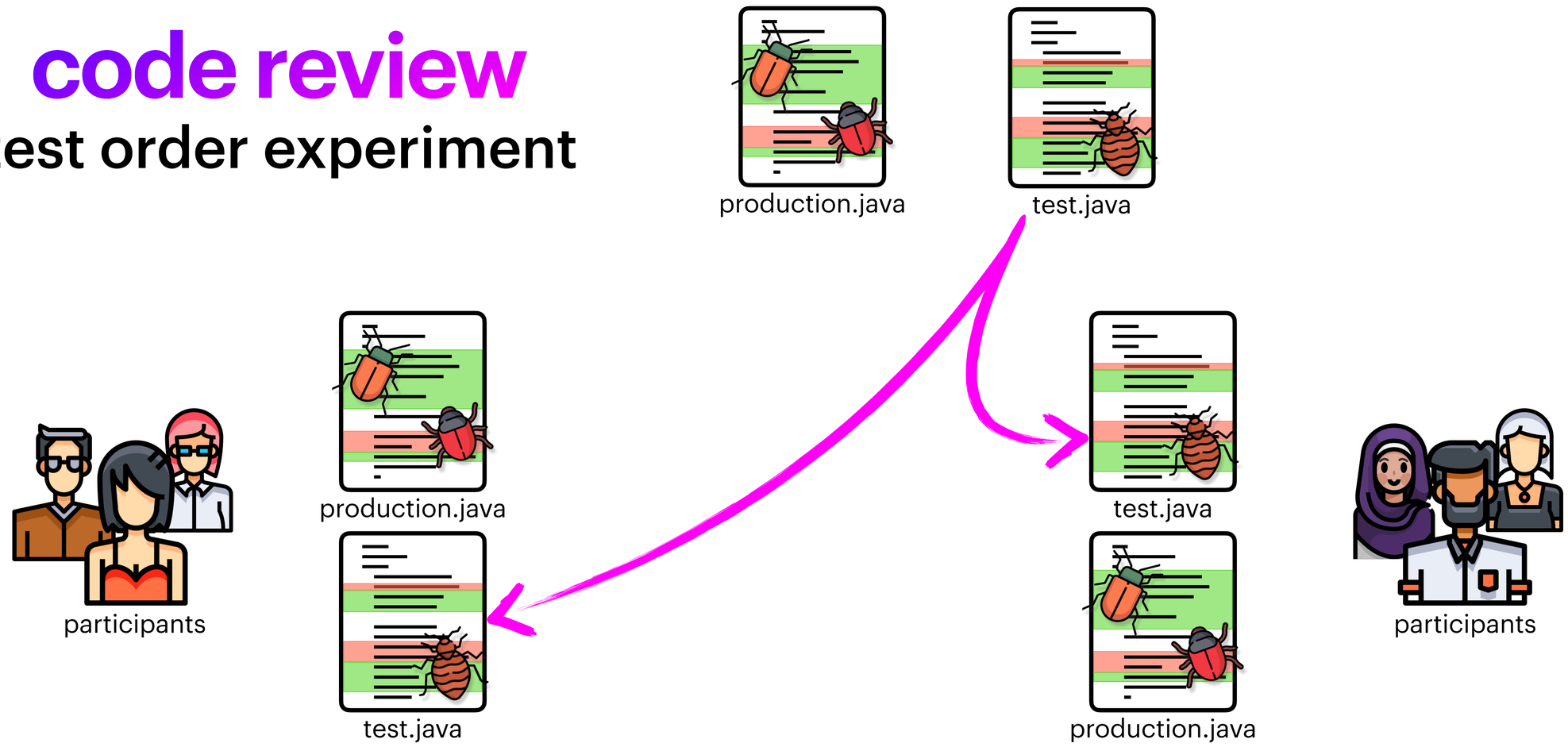
code review

test order experiment



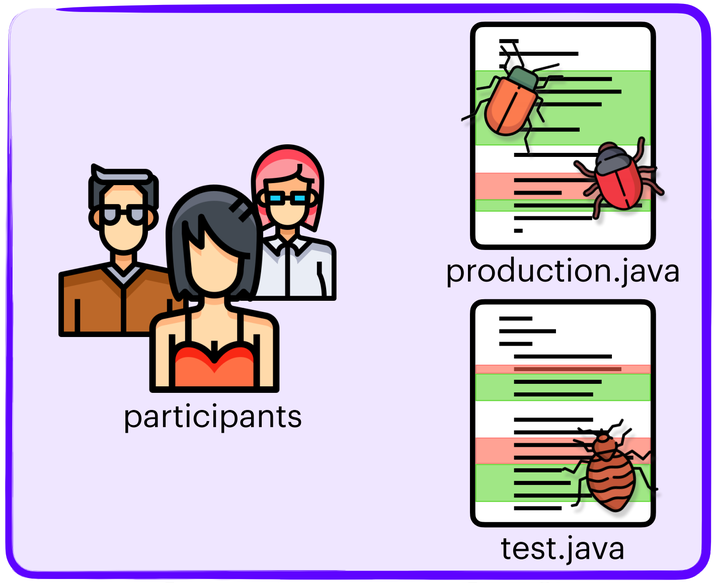
code review

test order experiment

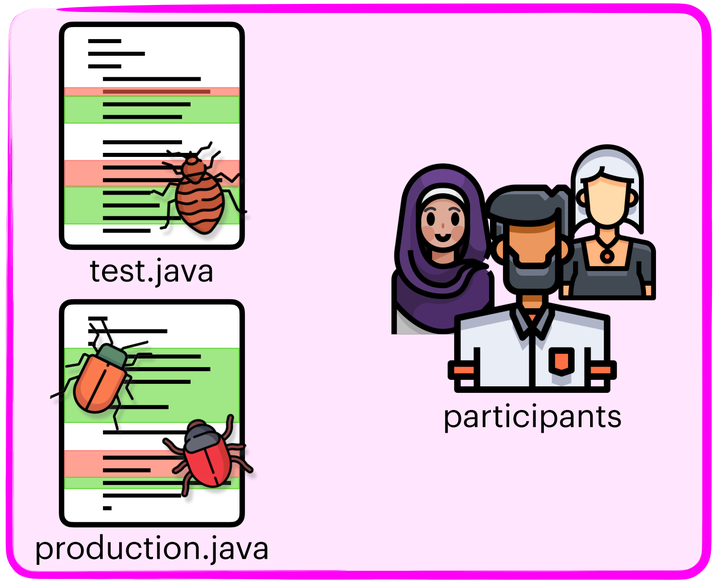


code review

test order experiment



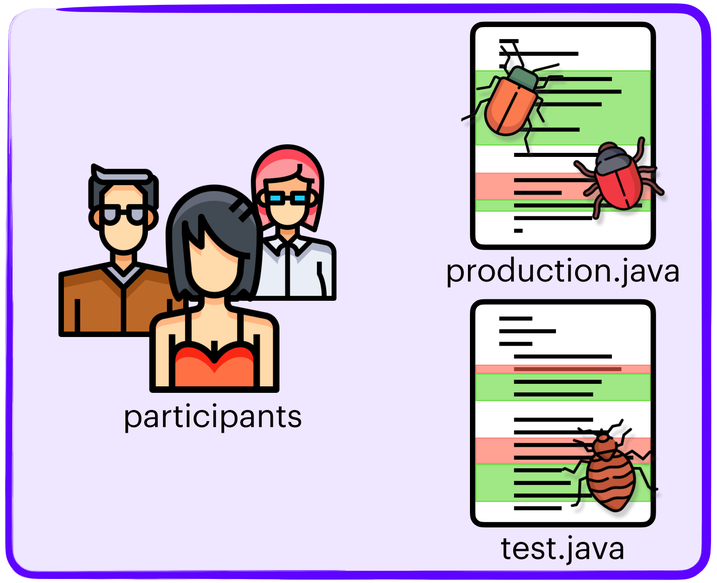
production code first order



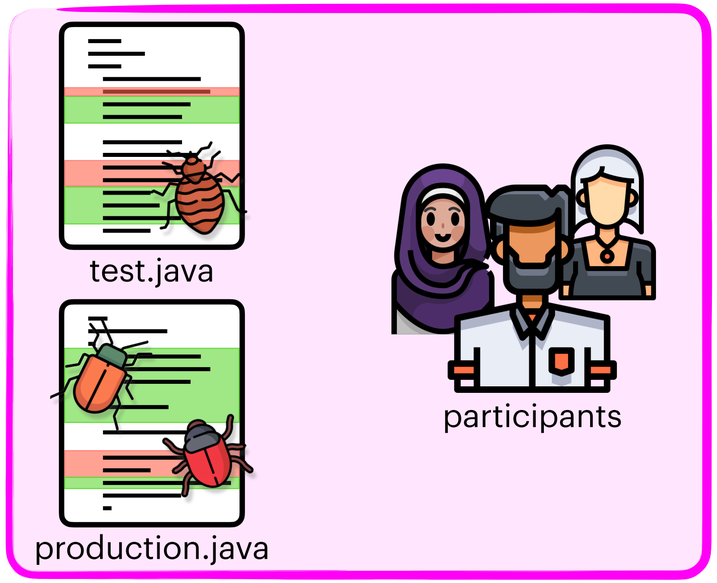
test code first order

code review

test order experiment

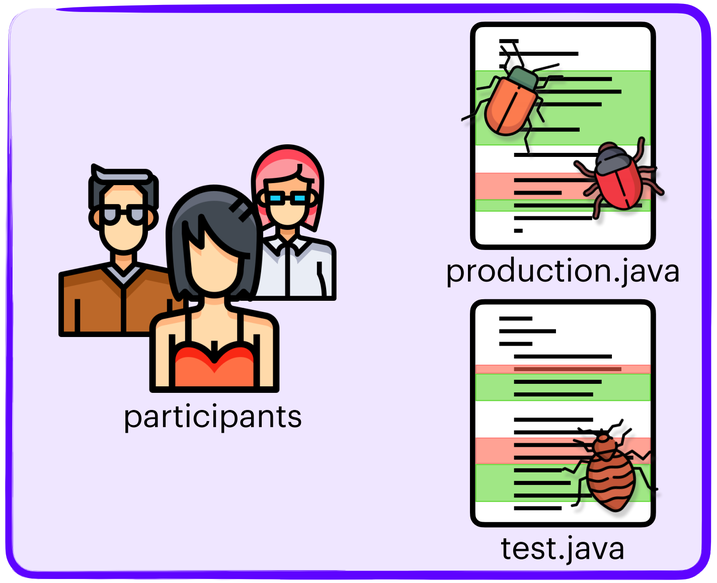


same likelihood of finding the production bugs



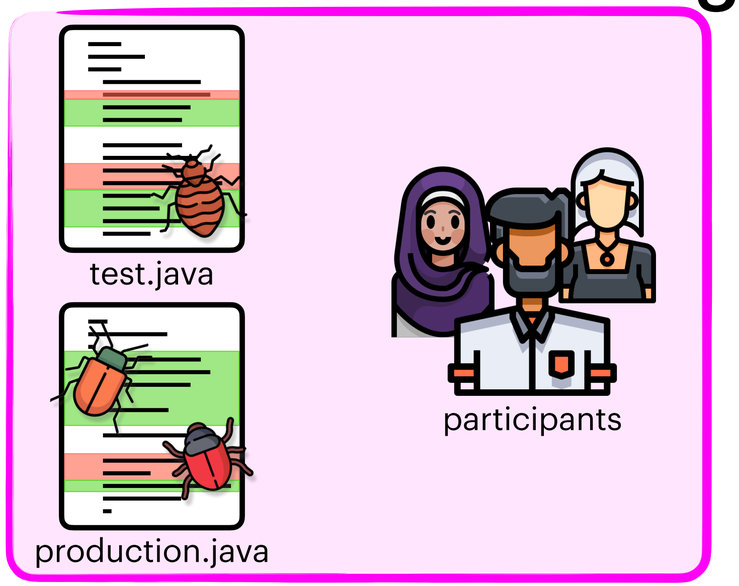
code review

test order experiment



same likelihood of finding the production bugs

250% more likely to find the test bug



code review

test files

Handle cancelling of remote que

https://github.com/github/vscode-codeq Work 110%

Conversation 14 | Commits 2 | Checks 22 | Files changed 5

Changes from all commits | File filter | Conversations | 0 / 5 files

Filter changed files

- extensions/ql-vscode/src
  - authentication.ts
  - query-history.ts
  - remote-queries
    - gh-actions-api-client.ts
    - remote-queries-manage...
- vscode-tests/no-workspace/r...
  - gh-actions-api-client.tes...

Beta Give feedback

11 extensions/ql-vscode/src/authentication.ts

```
@@ -7,7 +7,7 @@ const GITHUB_AUTH_PROVIDER_ID = 'github';
7  7   // https://docs.github.com/apps/building-oauth-apps/understand
8  8   const SCOPES = ['repo'];
9  9
10     - /**
   10  + /**
11 11   * Handles authentication to GitHub, using the VS Code [authen
12 12   */
13 13  export class Credentials {
@@ -18,6 +18,15 @@ export class Credentials {
18 18    // eslint-disable-next-line @typescript-eslint/no-empty-func
19 19    private constructor() { }
20 20
   21  +  /**
   22  +   * Initializes an instance of credentials with an octokit in
   23  +   *
   24  +   * Do not call this method until you know you actually need
   25  +   * since calling this method will require the user to log in
   26  +   *
   27  +   * @param context The extension context.
   28  +   * @returns An instance of credentials.
   29  +   */
21 30    static async initialize(context: vscode.ExtensionContext): P
22 31      const c = new Credentials();
23 32      c.registerListeners(context);
```

25 extensions/ql-vscode/src/query-history.ts

```
@@ -36,6 +36,8 @@ import { QueryStatus } from './query-status'
36 36  import { slurpQueryHistory, splatQueryHistory } from './query-
37 37  import * as fs from 'fs-extra';
38 38  import { CliVersionConstraint } from './cli';
   39  + import { Credentials } from './authentication';
   40  + import { cancelRemoteQuery } from './remote-queries/gh-actions
```

code review

production files

Handle cancelling of remote que

https://github.com/github/vscode-codeq

Conversation 14 | Commits 2 | Checks 22 | Files changed 5

Changes from all commits | File filter | Conversations

0 / 5 files

Filter changed files

- extensions/ql-vscode/src
  - authentication.ts
  - query-history.ts
  - remote-queries
    - gh-actions-api-client.ts
    - remote-queries-manage...
- vscode-tests/no-workspace/r...
  - gh-actions-api-client.tes...

Beta Give feedback

11 extensions/ql-vscode/src/authentication.ts

```
@@ -7,7 +7,7 @@ const GITHUB_AUTH_PROVIDER_ID = 'github';
 7  7  // https://docs.github.com/apps/building-oauth-apps/understand
 8  8  const SCOPES = ['repo'];
 9  9
10     - /**
   10  + /**
11 11   * Handles authentication to GitHub, using the VS Code [authen
12 12   */
13 13  export class Credentials {
@@ -18,6 +18,15 @@ export class Credentials {
18 18    // eslint-disable-next-line @typescript-eslint/no-empty-func
19 19    private constructor() { }
20 20
   21  +  /**
   22  +   * Initializes an instance of credentials with an octokit in
   23  +   *
   24  +   * Do not call this method until you know you actually need
   25  +   * since calling this method will require the user to log in
   26  +   *
   27  +   * @param context The extension context.
   28  +   * @returns An instance of credentials.
   29  +   */
21 30    static async initialize(context: vscode.ExtensionContext): P
22 31      const c = new Credentials();
23 32      c.registerListeners(context);
```

25 extensions/ql-vscode/src/query-history.ts

```
@@ -36,6 +36,8 @@ import { QueryStatus } from './query-status'
36 36  import { slurpQueryHistory, splatQueryHistory } from './query-
37 37  import * as fs from 'fs-extra';
38 38  import { CliVersionConstraint } from './cli';
   39  + import { Credentials } from './authentication';
   40  + import { cancelRemoteQuery } from './remote-queries/gh-actions
```

code review

production files

code review

production files

- We analyzed review comments for 200k pull requests from some 140 popular projects in GitHub...

code review

production files

- We analyzed review comments for 200k pull requests from some 140 popular projects in GitHub...
- ... and found something remarkable.

code review

production files

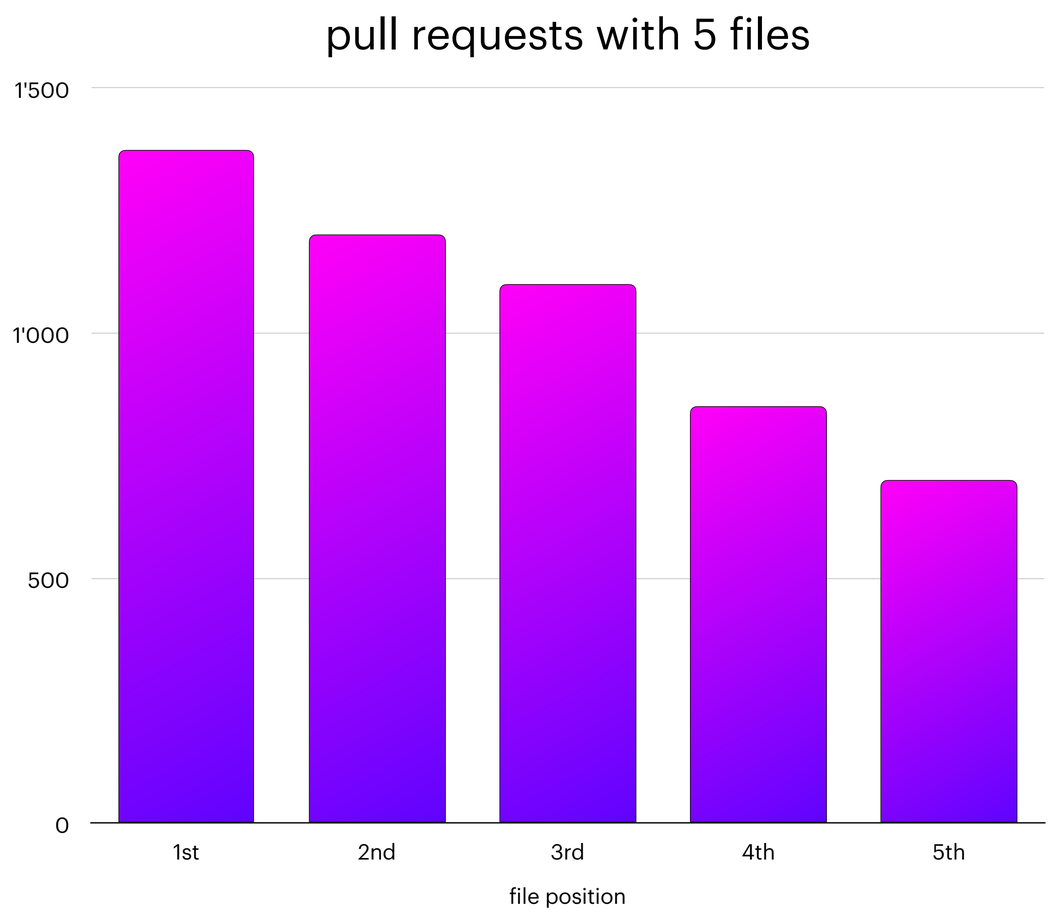
- We analyzed review comments for 200k pull requests from some 140 popular projects in GitHub...
- ... and found something remarkable.

cumulative number of review comments by file position

code review
production files

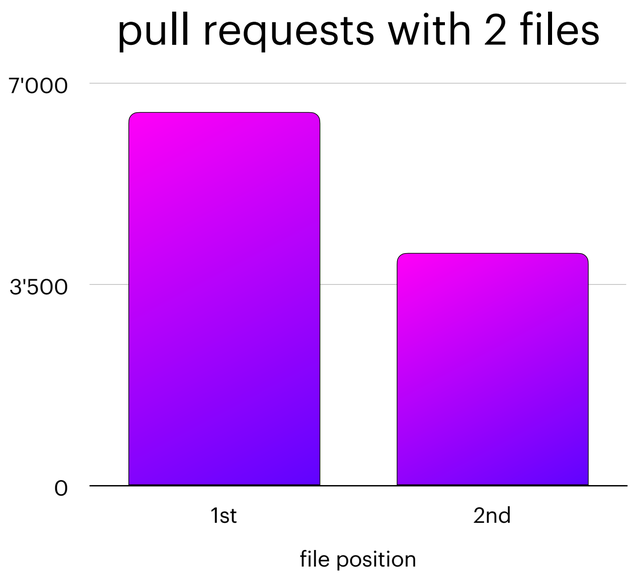
- We analyzed review comments for 200k pull requests from some 140 popular projects in GitHub...
- ... and found something remarkable.

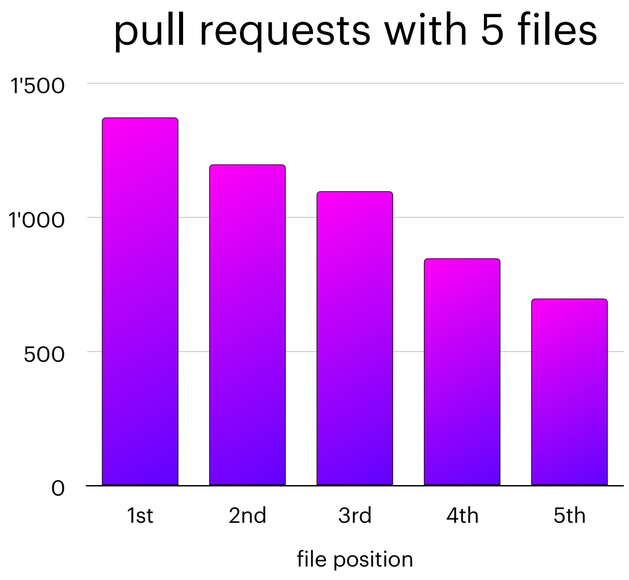
cumulative number of review comments by file position



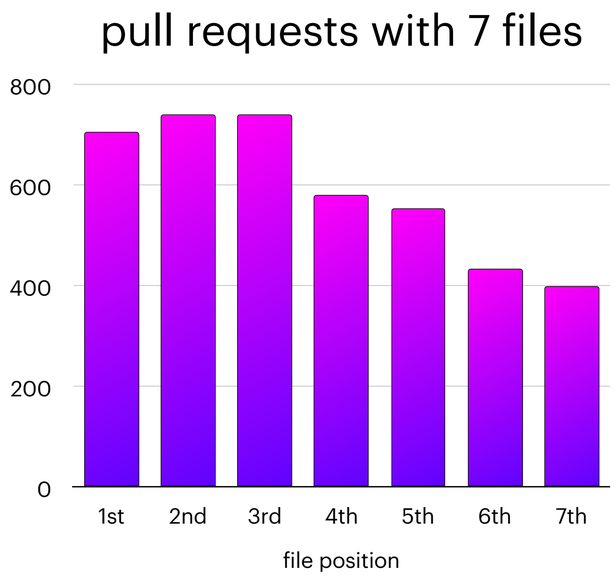
code review
production files

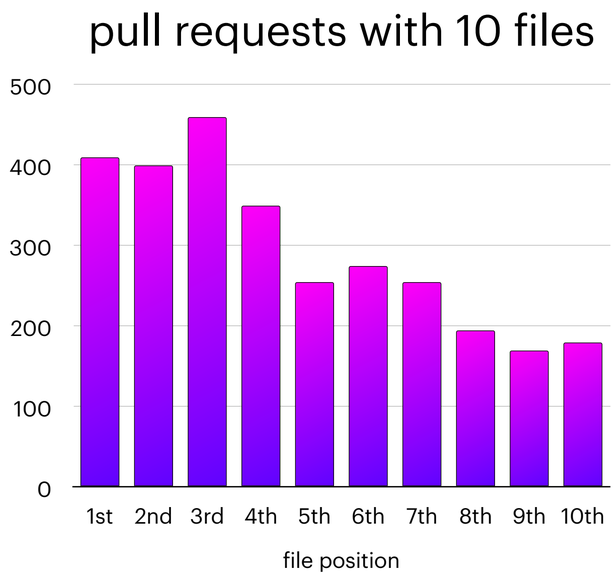
- We analyzed review comments for 200k pull requests from some 140 popular projects in GitHub...
- ... and found something remarkable.





cumulative number of review comments by file position



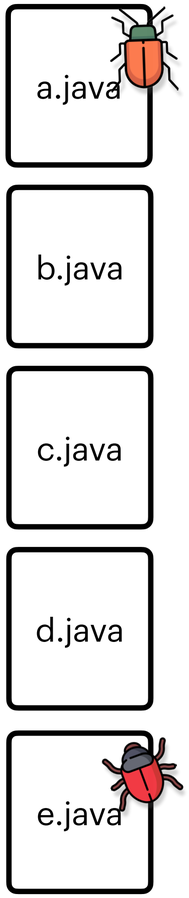


code review

production files experiment

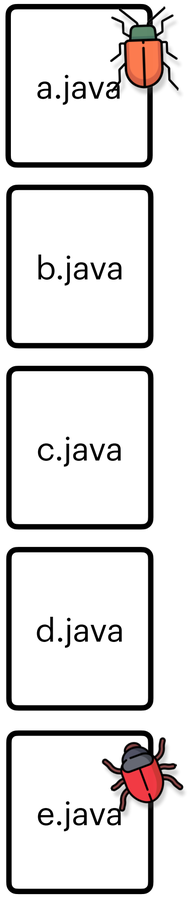
code review

production files experiment



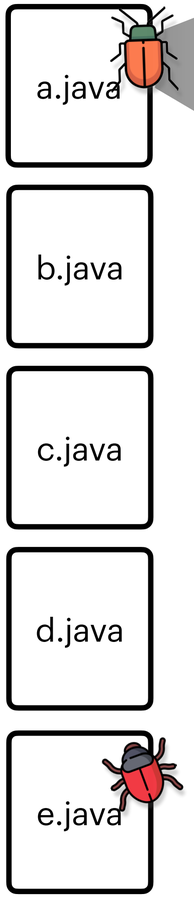
code review

production files experiment



code review

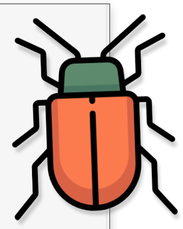
production files experiment




```
26          switch (destinationAddress.getCountry()) {
27              case "USA":
28                  shippingCost = shippingCost * 1.2;
29                  break;
30              case "Canada":
31                  shippingCost = shippingCost * 1.18;
32                  break;
33              case "Mexico":
34                  shippingCost = shippingCost * 1.35;
35                  break;
36              case "UK":
37                  shippingCost = shippingCost * 1.27;
```

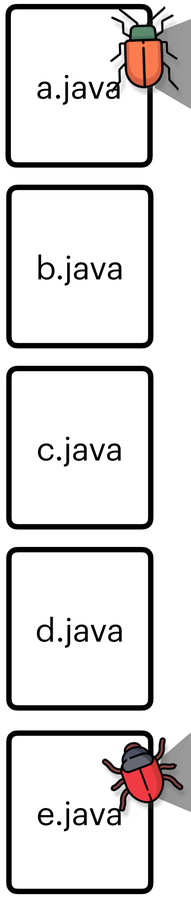
MB: Missing Break defect: Here a break statement is missing. In this way, when the country is UK, the execution will fall through the default case and a wrong tax of 1.27 * 2 will be applied.

```
38              default:
39                  shippingCost = shippingCost * 2;
40          }
41          return shippingCost;
```



code review

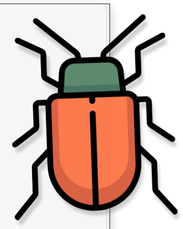
production files experiment




```
26          switch (destinationAddress.getCountry()) {
27              case "USA":
28                  shippingCost = shippingCost * 1.2;
29                  break;
30              case "Canada":
31                  shippingCost = shippingCost * 1.18;
32                  break;
33              case "Mexico":
34                  shippingCost = shippingCost * 1.35;
35                  break;
36              case "UK":
37                  shippingCost = shippingCost * 1.27;
```

MB: Missing Break defect: Here a break statement is missing. In this way, when the country is UK, the execution will fall through the default case and a wrong tax of 1.27 * 2 will be applied.

```
38              default:
39                  shippingCost = shippingCost * 2;
40          }
41          return shippingCost;
```



```
7       /**
8        * Returns the discount rate based on the membership level of the
    customer.
9        * Customers at level 1 do not receive any discount.
10       * Customers at level 2 to 4 receive a 10% discount.
11       * Customers from level 5 included receive a 25% discount.
12       * @param membershipLevel - the level of membership of the customer
13       * @return the discount rate applied to the customer
14       */
15      public double getSaleDiscountRate(int membershipLevel){
16          double discountRate = 0;
17          if(membershipLevel > 2 && membershipLevel < 5) {
```

CC: Corner Case defect: Here the if statement is missing a check for the condition where customer.membershipLevel == 2. According to the Javadoc of the function, customers with membership level equal to 2 should receive a 10% discount

```
18              discountRate = 0.1;
19          }
20          if(membershipLevel >= 5) {
21              discountRate = 0.25;
22          }
23          return discountRate;
24      }
```

code review

production files experiment

```
26        switch (destinationAddress.getCountry()) {
27            case "USA":
28                shippingCost = shippingCost * 1.2;
29                break;
30            case "Canada":
31                shippingCost = shippingCost * 1.18;
32                break;
33            case "Mexico":
34                shippingCost = shippingCost * 1.35;
35                break;
36            case "UK":
37                shippingCost = shippingCost * 1.27;
```

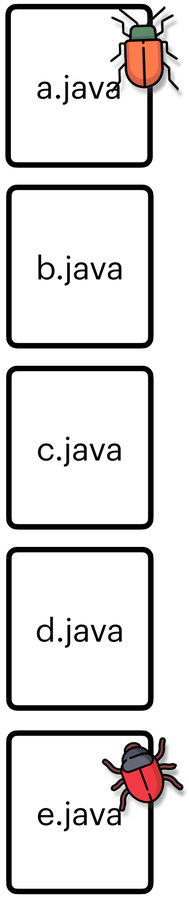
MB: Missing Break defect: Here a break statement is missing. In this way, when the country is UK, the execution will fall through the default case and a wrong tax of 1.27 * 2 will be applied.

```
38            default:
39                shippingCost = shippingCost * 2;
40        }
41        return shippingCost;
```

```
7     /**
8      * Returns the discount rate based on the membership level of the
customer.
9      * Customers at level 1 do not receive any discount.
10     * Customers at level 2 to 4 receive a 10% discount.
11     * Customers from level 5 included receive a 25% discount.
12     * @param membershipLevel - the level of membership of the customer
13     * @return the discount rate applied to the customer
14     */
15    public double getSaleDiscountRate(int membershipLevel){
16        double discountRate = 0;
17        if(membershipLevel > 2 && membershipLevel < 5) {
```

CC: Corner Case defect: Here the if statement is missing a check for the condition where customer.membershipLevel == 2. According to the Javadoc of the function, customers with membership level equal to 2 should receive a 10% discount

```
18            discountRate = 0.1;
19        }
20        if(membershipLevel >= 5) {
21            discountRate = 0.25;
22        }
23        return discountRate;
24    }
```



code review

production files experiment

```
26        switch (destinationAddress.getCountry()) {
27            case "USA":
28                shippingCost = shippingCost * 1.2;
29                break;
30            case "Canada":
31                shippingCost = shippingCost * 1.18;
32                break;
33            case "Mexico":
34                shippingCost = shippingCost * 1.35;
35                break;
36            case "UK":
37                shippingCost = shippingCost * 1.27;
```

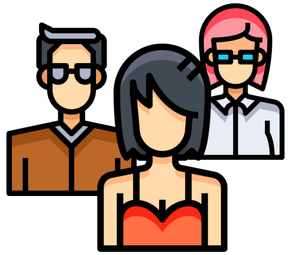
MB: Missing Break defect: Here a break statement is missing. In this way, when the country is UK, the execution will fall through the default case and a wrong tax of 1.27 * 2 will be applied.

```
38            default:
39                shippingCost = shippingCost * 2;
40        }
41        return shippingCost;
```

```
7     /**
8      * Returns the discount rate based on the membership level of the customer.
9      * Customers at level 1 do not receive any discount.
10     * Customers at level 2 to 4 receive a 10% discount.
11     * Customers from level 5 included receive a 25% discount.
12     * @param membershipLevel - the level of membership of the customer
13     * @return the discount rate applied to the customer
14     */
15    public double getSaleDiscountRate(int membershipLevel){
16        double discountRate = 0;
17        if(membershipLevel > 2 && membershipLevel < 5) {
```

CC: Corner Case defect: Here the if statement is missing a check for the condition where customer.membershipLevel == 2. According to the Javadoc of the function, customers with membership level equal to 2 should receive a 10% discount

```
18            discountRate = 0.1;
19        }
20        if(membershipLevel >= 5) {
21            discountRate = 0.25;
22        }
23        return discountRate;
24    }
```



participants

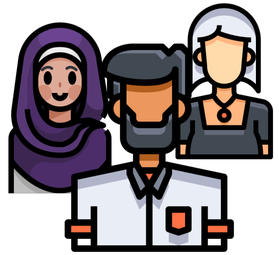
a.java

b.java

c.java

d.java

e.java



participants

code review

production files experiment

```
26        switch (destinationAddress.getCountry()) {
27            case "USA":
28                shippingCost = shippingCost * 1.2;
29                break;
30            case "Canada":
31                shippingCost = shippingCost * 1.18;
32                break;
33            case "Mexico":
34                shippingCost = shippingCost * 1.35;
35                break;
36            case "UK":
37                shippingCost = shippingCost * 1.27;
```

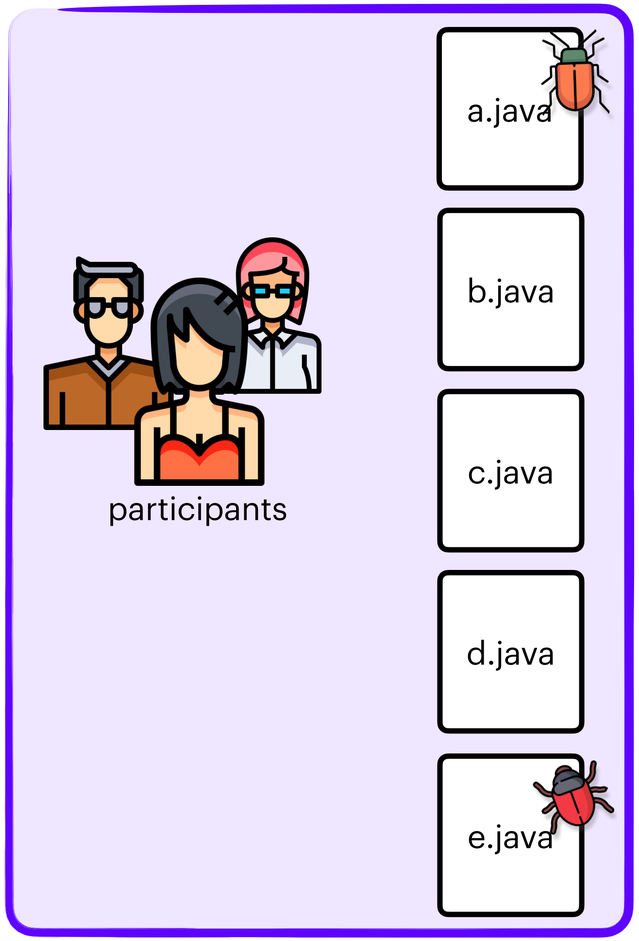
MB: Missing Break defect: Here a break statement is missing. In this way, when the country is UK, the execution will fall through the default case and a wrong tax of 1.27 * 2 will be applied.

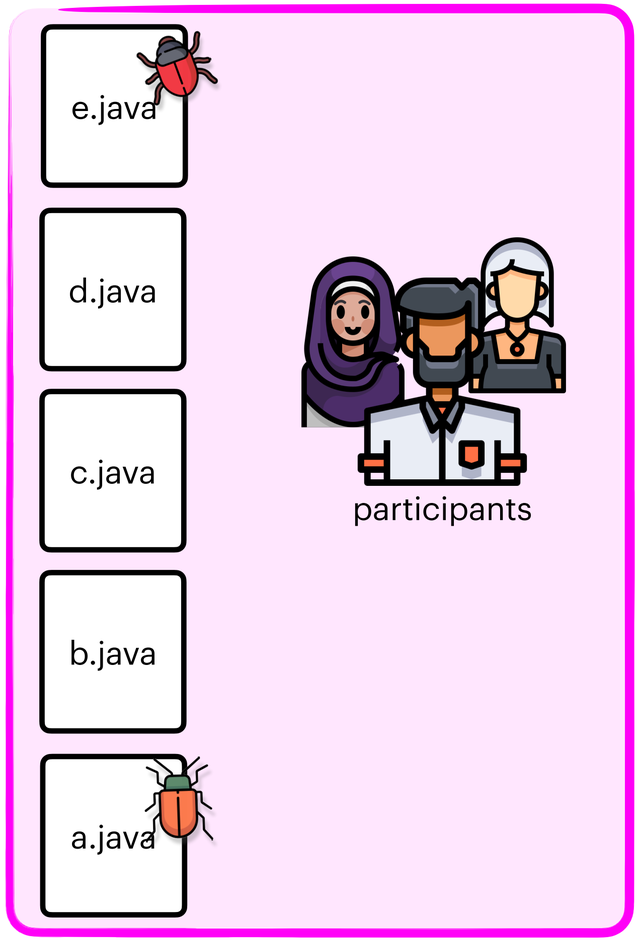
```
38            default:
39                shippingCost = shippingCost * 2;
40        }
41        return shippingCost;
```

```
7     /**
8      * Returns the discount rate based on the membership level of the
customer.
9      * Customers at level 1 do not receive any discount.
10     * Customers at level 2 to 4 receive a 10% discount.
11     * Customers from level 5 included receive a 25% discount.
12     * @param membershipLevel - the level of membership of the customer
13     * @return the discount rate applied to the customer
14     */
15    public double getSaleDiscountRate(int membershipLevel){
16        double discountRate = 0;
17        if(membershipLevel > 2 && membershipLevel < 5) {
```

CC: Corner Case defect: Here the if statement is missing a check for the condition where customer.membershipLevel == 2. According to the Javadoc of the function, customers with membership level equal to 2 should receive a 10% discount

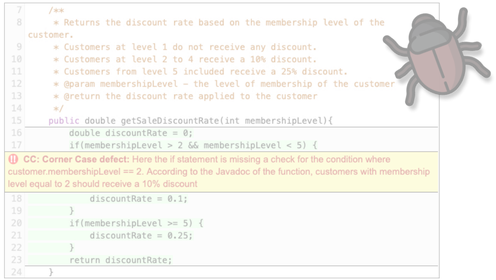
```
18            discountRate = 0.1;
19        }
20        if(membershipLevel >= 5) {
21            discountRate = 0.25;
22        }
23        return discountRate;
24    }
```





code review

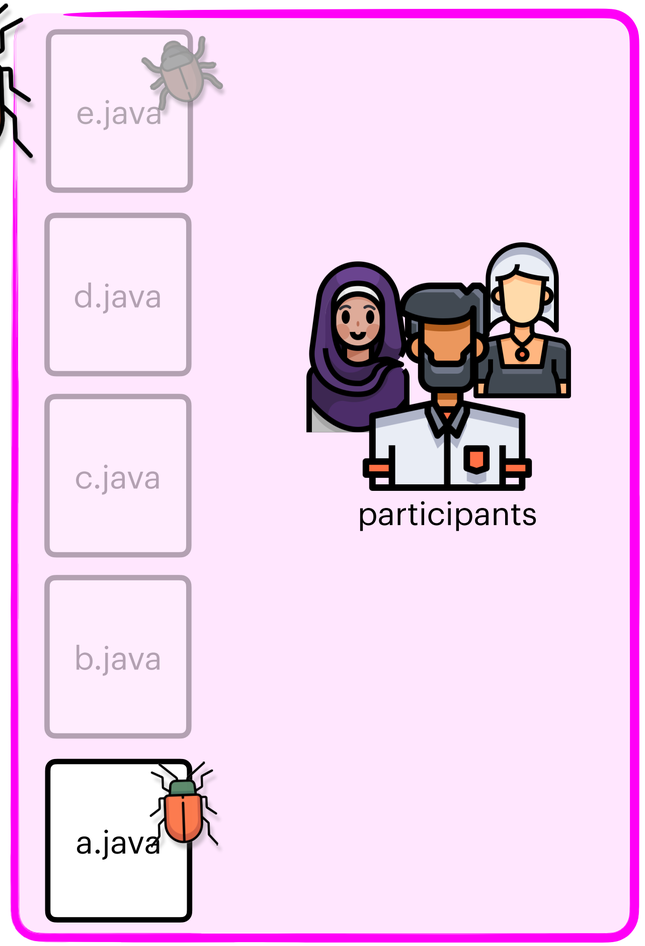
production files experiment




```
26        switch (destinationAddress.getCountry()) {
27            case "USA":
28                shippingCost = shippingCost * 1.2;
29                break;
30            case "Canada":
31                shippingCost = shippingCost * 1.18;
32                break;
33            case "Mexico":
34                shippingCost = shippingCost * 1.35;
35                break;
36            case "UK":
37                shippingCost = shippingCost * 1.27;
```

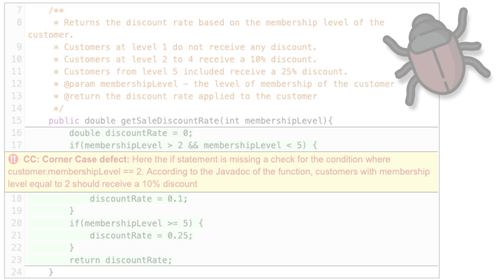
MB: Missing Break defect: Here a break statement is missing. In this way, when the country is UK, the execution will fall through the default case and a wrong tax of 1.27 * 2 will be applied.

```
38            default:
39                shippingCost = shippingCost * 2;
40        }
41        return shippingCost;
```



code review

production files experiment

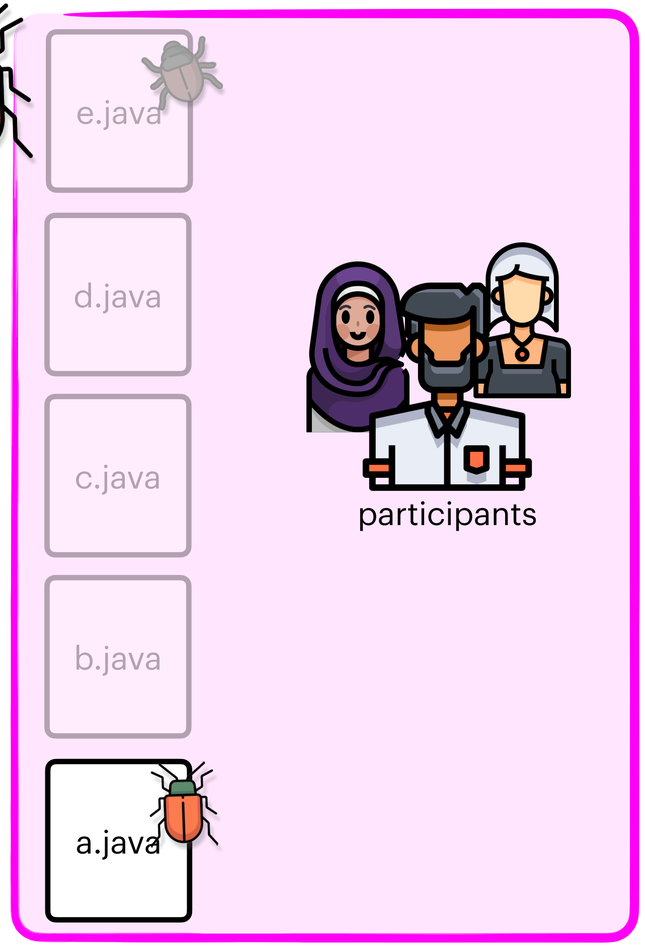



```
26        switch (destinationAddress.getCountry()) {
27            case "USA":
28                shippingCost = shippingCost * 1.2;
29                break;
30            case "Canada":
31                shippingCost = shippingCost * 1.18;
32                break;
33            case "Mexico":
34                shippingCost = shippingCost * 1.35;
35                break;
36            case "UK":
37                shippingCost = shippingCost * 1.27;
```

MB: Missing Break defect: Here a break statement is missing. In this way, when the country is UK, the execution will fall through the default case and a wrong tax of 1.27 * 2 will be applied.

```
38            default:
39                shippingCost = shippingCost * 2;
40        }
41        return shippingCost;
```

same likelihood of finding the bug



code review

production files experiment

```
26        switch (destinationAddress.getCountry()) {
27            case "USA":
28                shippingCost = shippingCost * 1.2;
29                break;
30            case "Canada":
31                shippingCost = shippingCost * 1.18;
32                break;
33            case "Mexico":
34                shippingCost = shippingCost * 1.35;
35                break;
36            case "UK":
37                shippingCost = shippingCost * 1.27;
```

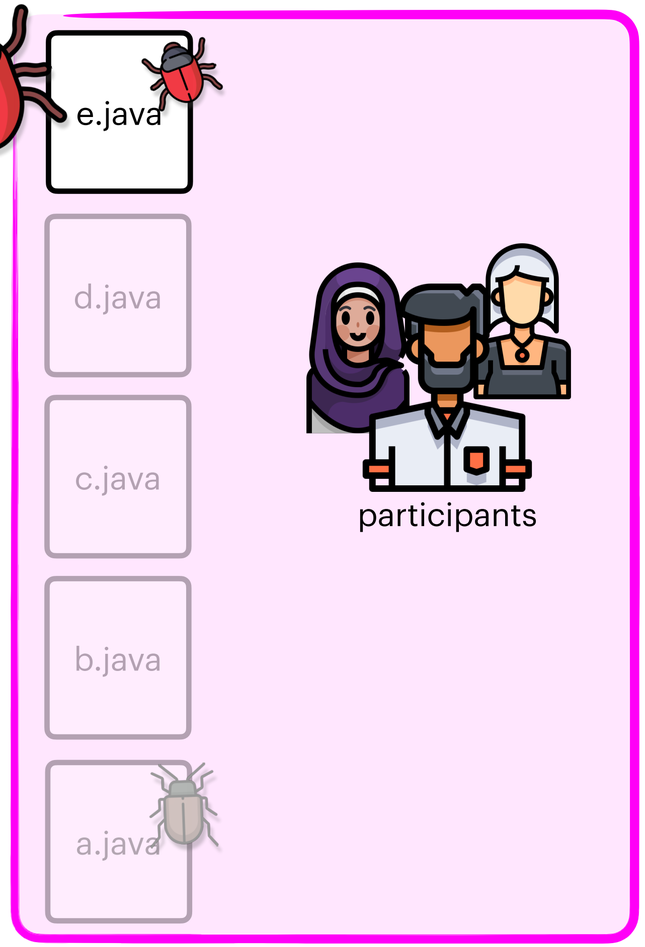
MB: Missing Break defect: Here a break statement is missing. In this way, when the country is UK, the execution will fall through the default case and a wrong tax of 1.27 * 2 will be applied.

```
38            default:
39                shippingCost = shippingCost * 2;
40        }
41        return shippingCost;
```


```
7     /**
8      * Returns the discount rate based on the membership level of the customer.
9      * Customers at level 1 do not receive any discount.
10     * Customers at level 2 to 4 receive a 10% discount.
11     * Customers from level 5 included receive a 25% discount.
12     * @param membershipLevel - the level of membership of the customer
13     * @return the discount rate applied to the customer
14     */
15    public double getSaleDiscountRate(int membershipLevel){
16        double discountRate = 0;
17        if(membershipLevel > 2 && membershipLevel < 5) {
```

CC: Corner Case defect: Here the if statement is missing a check for the condition where customer.membershipLevel == 2. According to the Javadoc of the function, customers with membership level equal to 2 should receive a 10% discount

```
18            discountRate = 0.1;
19        }
20        if(membershipLevel >= 5) {
21            discountRate = 0.25;
22        }
23        return discountRate;
24    }
```



code review

production files experiment

```
26          switch (destinationAddress.getCountry()) {
27              case "USA":
28                  shippingCost = shippingCost * 1.2;
29                  break;
30              case "Canada":
31                  shippingCost = shippingCost * 1.18;
32                  break;
33              case "Mexico":
34                  shippingCost = shippingCost * 1.35;
35                  break;
36              case "UK":
37                  shippingCost = shippingCost * 1.27;
```

MB: Missing Break defect: Here a break statement is missing. In this way, when the country is UK, the execution will fall through the default case and a wrong tax of 1.27 * 2 will be applied.

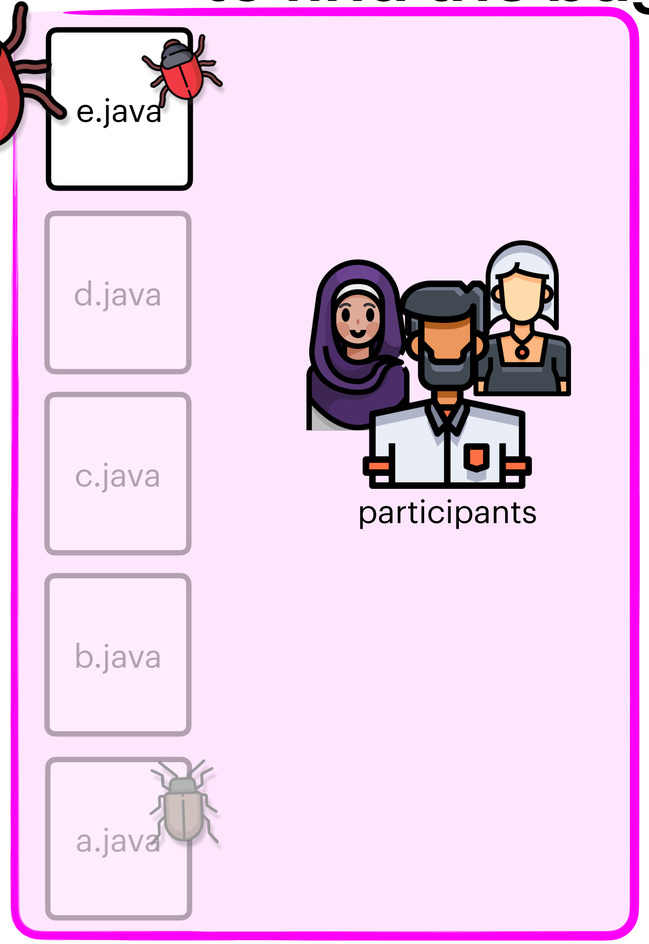
```
38              default:
39                  shippingCost = shippingCost * 2;
40          }
41          return shippingCost;
```

175% more likely
to find the bug


```
7       /**
8        * Returns the discount rate based on the membership level of the customer.
9        * Customers at level 1 do not receive any discount.
10       * Customers at level 2 to 4 receive a 10% discount.
11       * Customers from level 5 included receive a 25% discount.
12       * @param membershipLevel - the level of membership of the customer
13       * @return the discount rate applied to the customer
14       */
15      public double getSaleDiscountRate(int membershipLevel){
16          double discountRate = 0;
17          if(membership Level > 2 && membershipLevel < 5) {
```

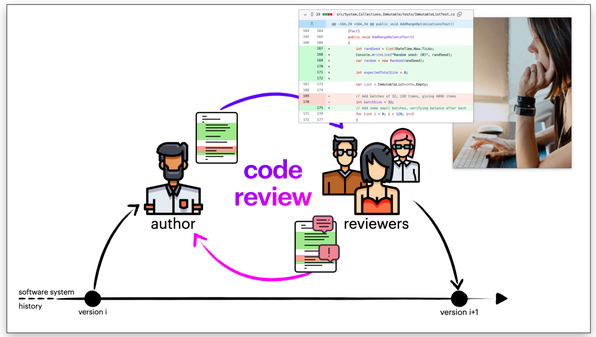
CC: Corner Case defect: Here the if statement is missing a check for the condition where customer.membershipLevel == 2. According to the Javadoc of the function, customers with membership level equal to 2 should receive a 10% discount

```
18              discountRate = 0.1;
19          }
20          if(membershipLevel >= 5) {
21              discountRate = 0.25;
22          }
23          return discountRate;
24      }
```

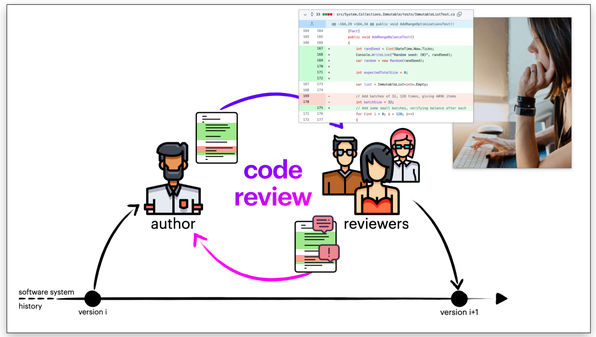


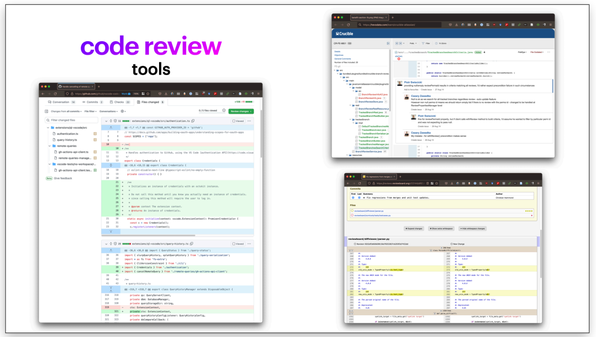
review

review

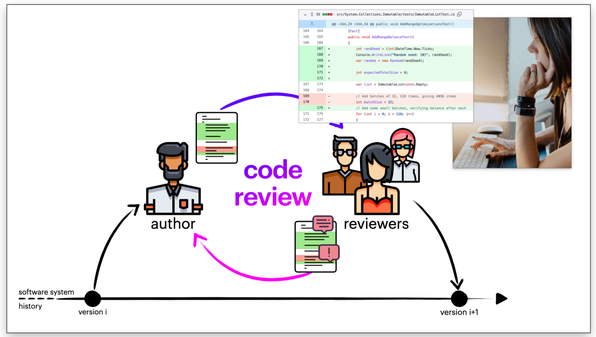

code
review
author
reviewers
software system
history
version i
version i+1

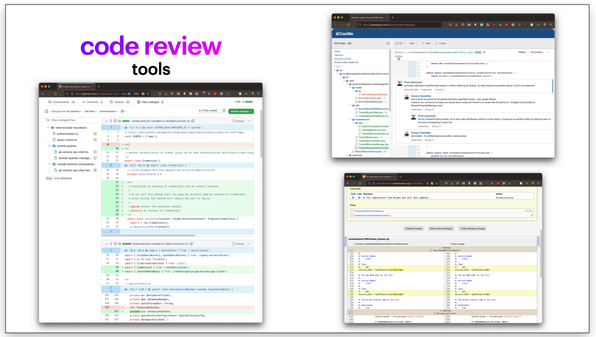
review


code
review
author
reviewers
software system
history
version i
version i+1


code review
tools

review

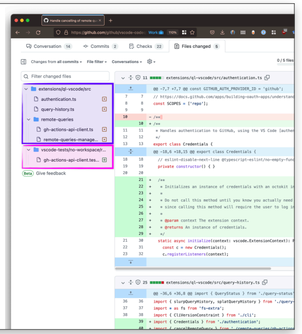




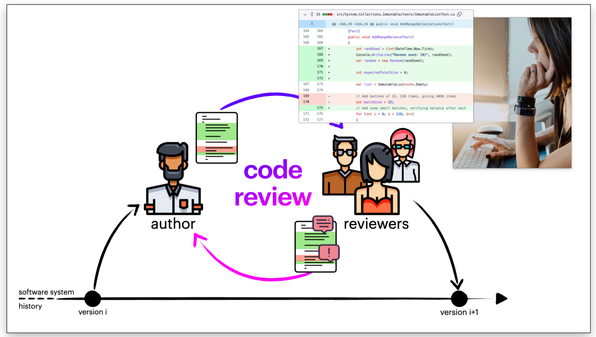
code review
test files

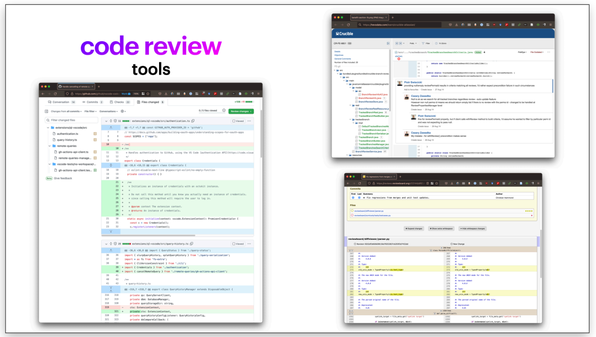
- Developers see test code as less important than production code.
- Test files are almost twice less likely to be discussed during code review when together with production files.

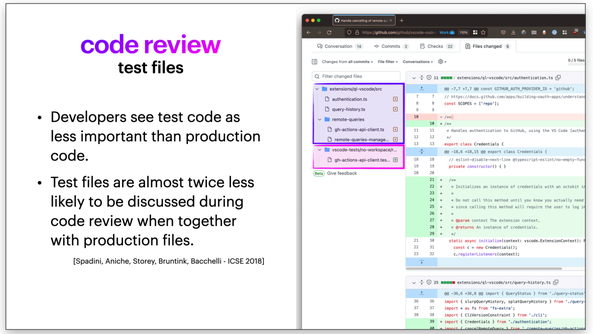
[Spadini, Aniche, Storey, Bruntink, Bacchelli - ICSE 2018]

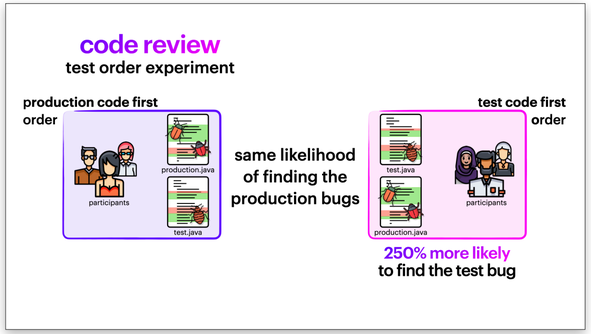


review

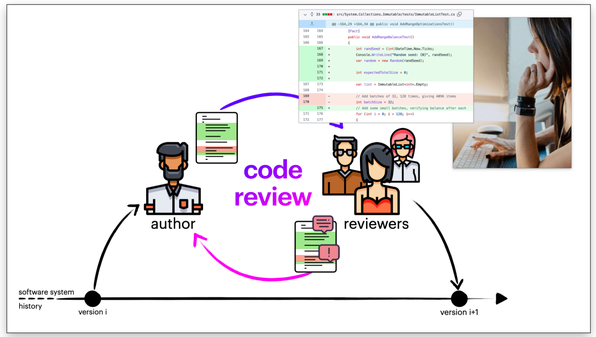

code review
author
reviewers
software system history
version i
version i+1


code review
tools

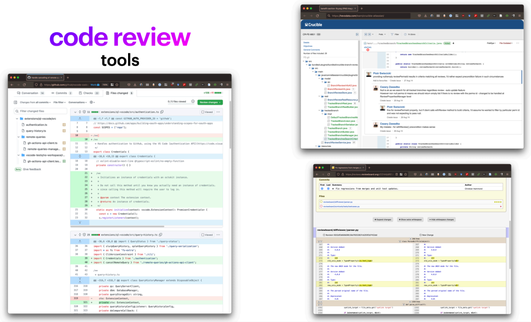

code review
test files
• Developers see test code as less important than production code.
• Test files are almost twice less likely to be discussed during code review when together with production files.
[Spadini, Aniche, Storey, Bruntink, Bacchelli - ICSE 2018]


code review
test order experiment
production code first order
participants
production.java
test.java
same likelihood of finding the production bugs
test code first order
test.java
participants
production.java
250% more likely
to find the test bug

review



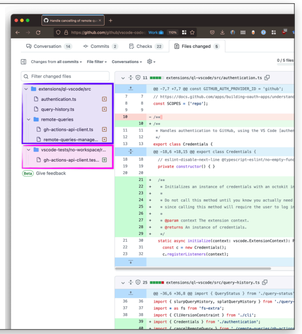
code review
tools



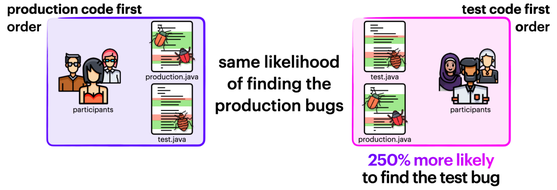
code review
test files

- Developers see test code as less important than production code.
- Test files are almost twice less likely to be discussed during code review when together with production files.

[Spadini, Aniche, Storey, Bruntink, Bacchelli - ICSE 2018]



code review
test order experiment

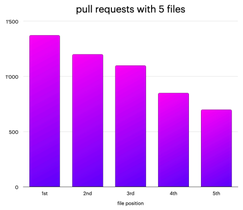


code review
production files

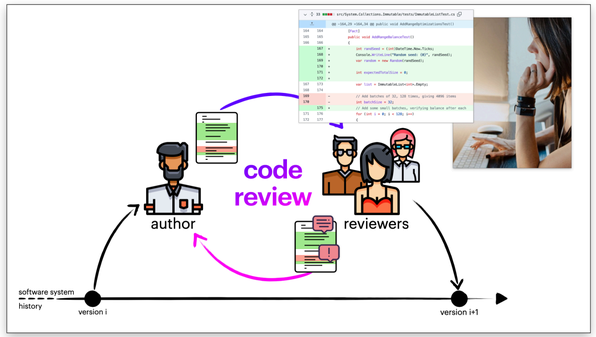
- We analyzed review comments for 200k pull requests from 138 popular projects in GitHub...
- ... and found something remarkable.

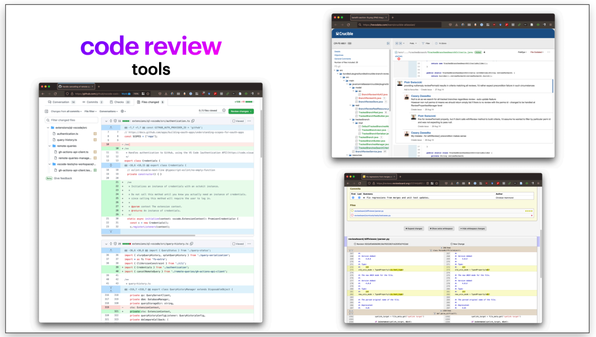
[Fregnan, Braz, D'Ambros, Çalikli, Bacchelli]

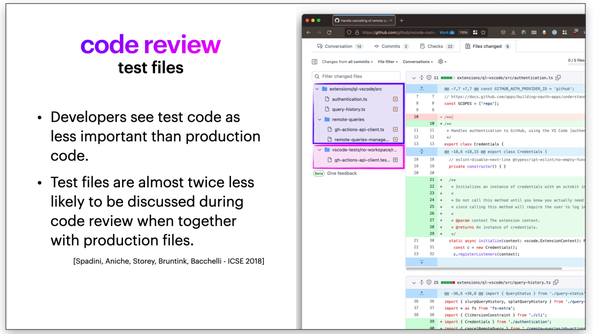
cumulative number of review comments by file position

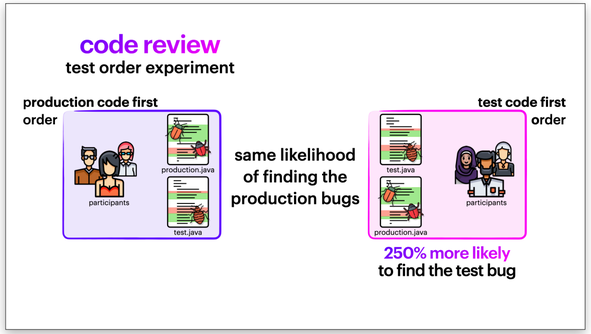


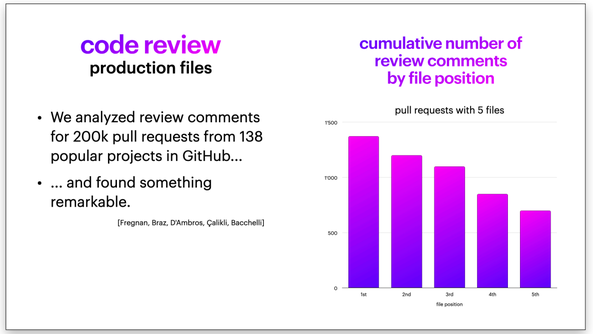
review

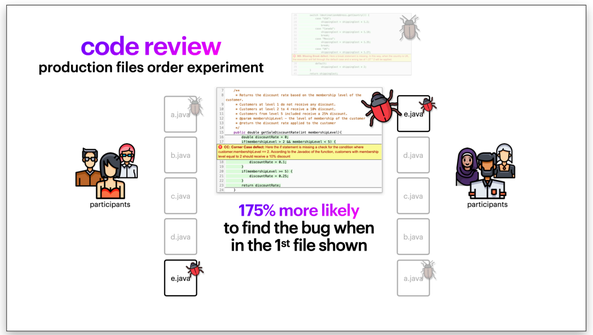

code review
author
reviewers
software system history
version i
version i+1


code review
tools

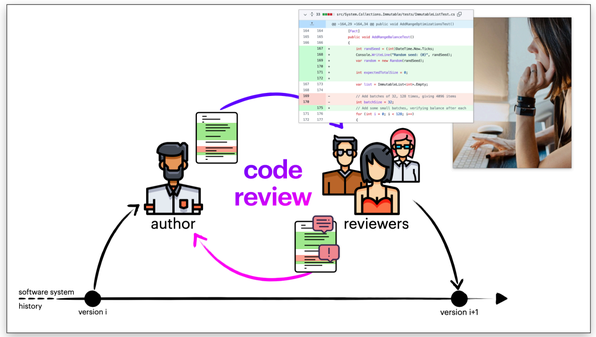

code review
test files
• Developers see test code as less important than production code.
• Test files are almost twice less likely to be discussed during code review when together with production files.
[Spadini, Aniche, Storey, Bruntink, Bacchelli - ICSE 2018]

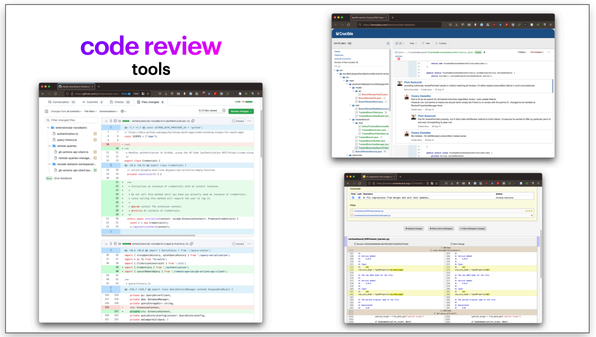

code review
test order experiment
production code first order
test code first order
participants
production.java
test.java
same likelihood of finding the production bugs
250% more likely to find the test bug


code review
production files
• We analyzed review comments for 200k pull requests from 138 popular projects in GitHub...
• ... and found something remarkable.
[Fregnan, Braz, D'Ambros, Çalikli, Bacchelli]
cumulative number of review comments by file position
pull requests with 5 files
1500
1000
500
0
1st
2nd
3rd
4th
5th
file position


code review
production files order experiment
participants
a.java
b.java
c.java
d.java
e.java
175% more likely to find the bug when in the 1st file shown

review

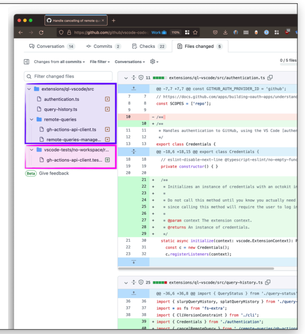




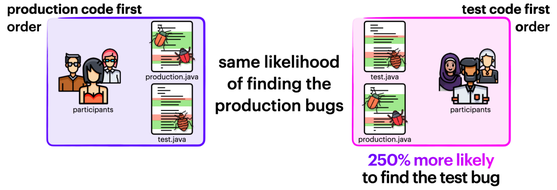
code review
test files

- Developers see test code as less important than production code.
- Test files are almost twice less likely to be discussed during code review when together with production files.

[Spadini, Aniche, Storey, Bruntink, Bacchelli - ICSE 2018]



code review
test order experiment

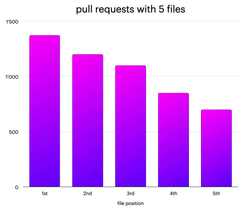


code review
production files

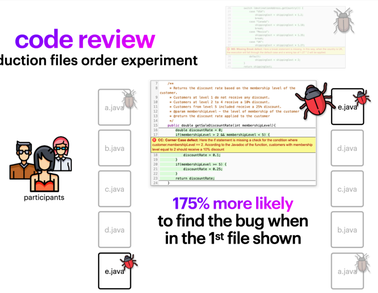
- We analyzed review comments for 200k pull requests from 138 popular projects in GitHub...
- ... and found something remarkable.

[Fregnan, Braz, D'Ambros, Çalikli, Bacchelli]

cumulative number of review comments by file position

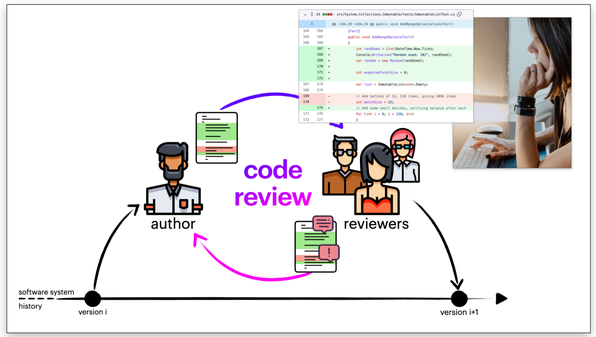


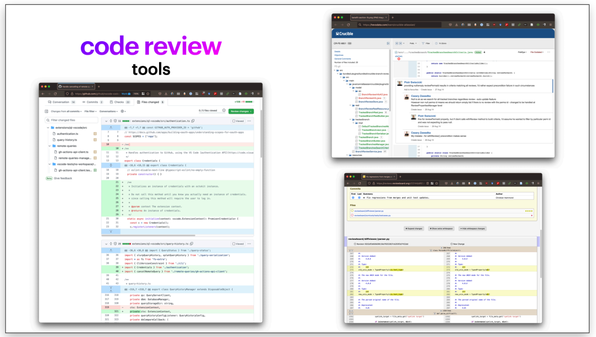
code review
production files order experiment

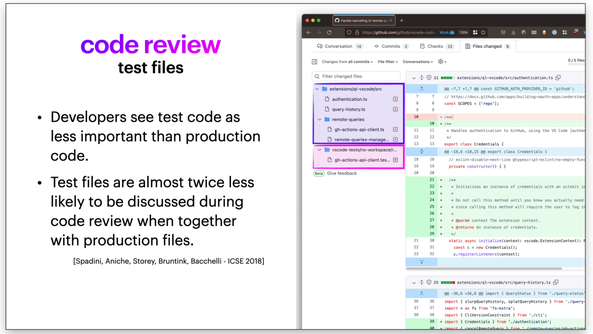


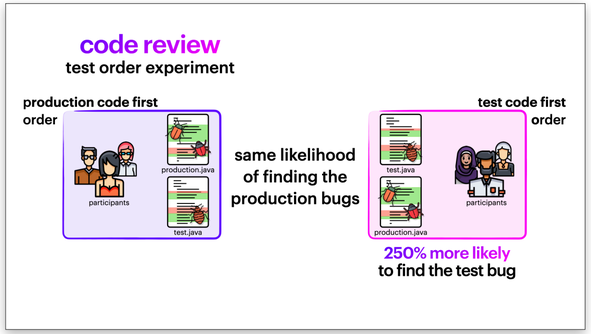
takeaways

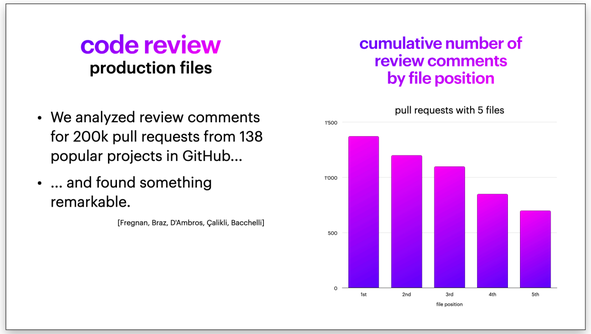
review

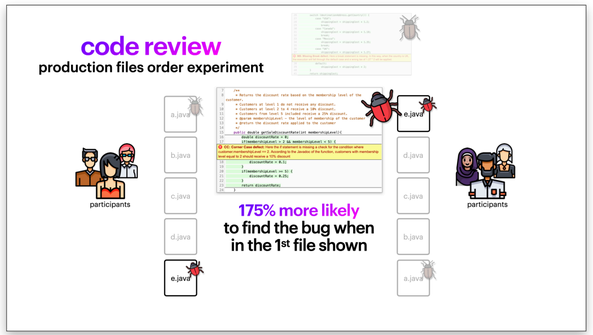








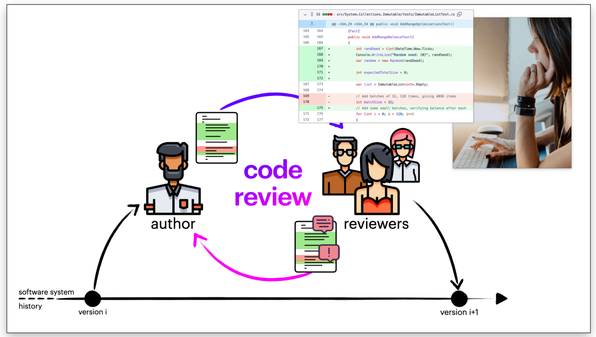


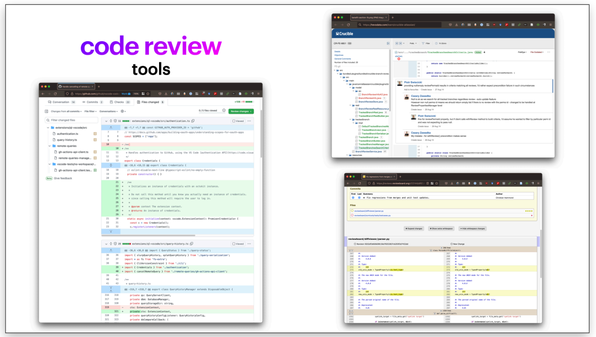


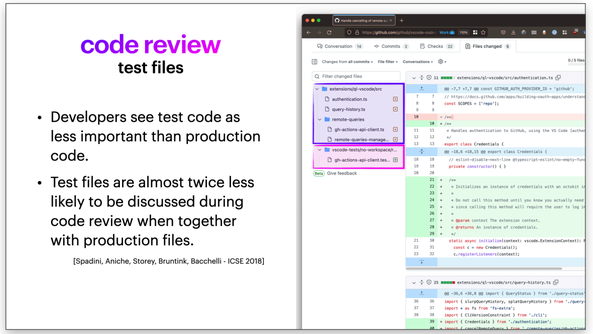
takeaways

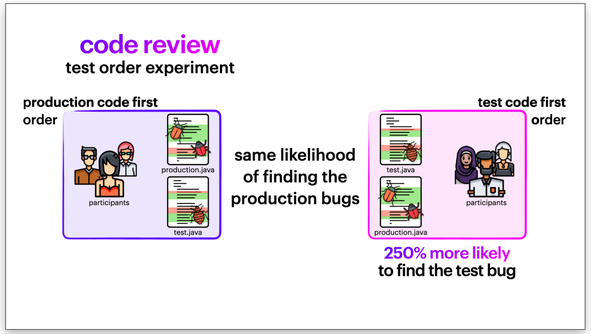
- **reviewers**: be aware of this effect and decide where to start your review in a principled way.

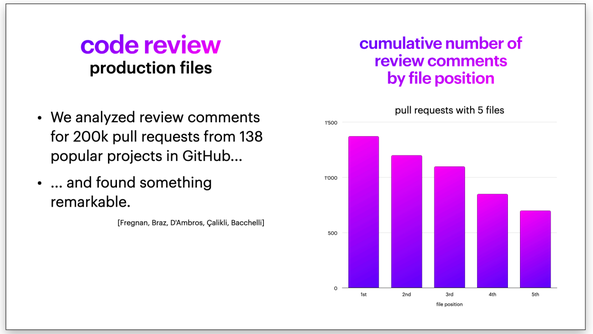
review

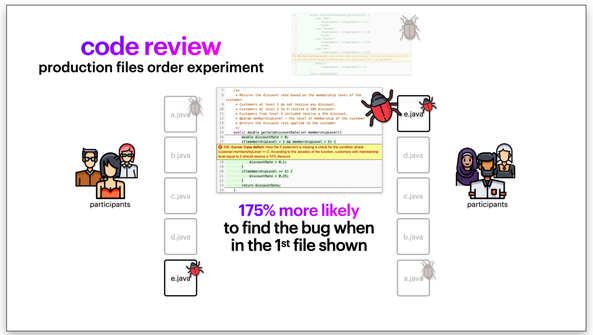








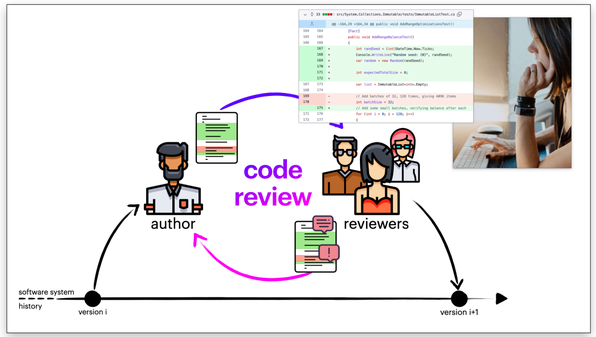


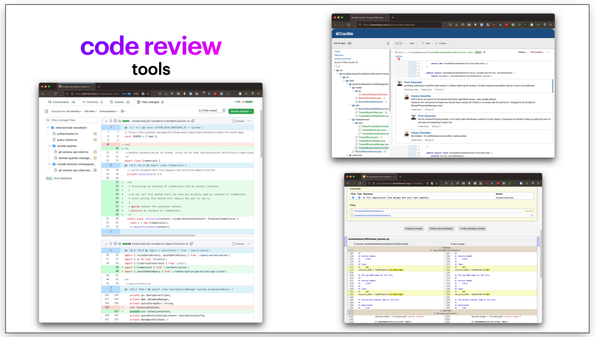


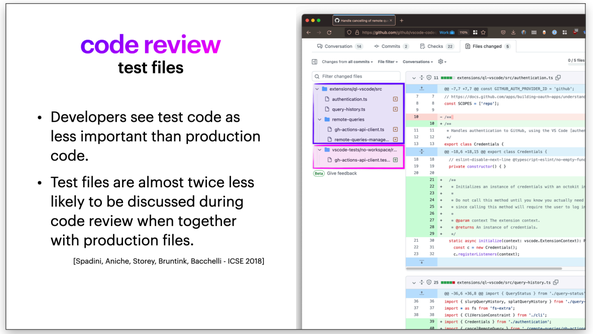
takeaways

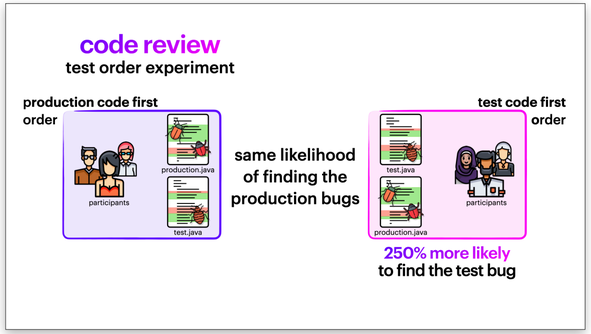
- **reviewers**: be aware of this effect and decide where to start your review in a principled way.
- **authors**: guide the reviewers towards the most challenging part of your change.

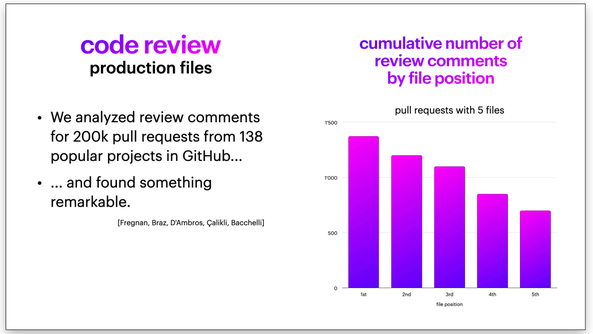
review

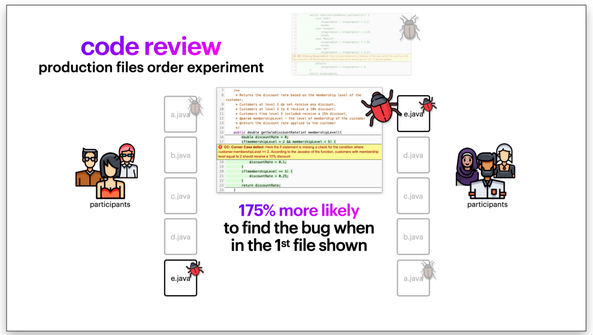








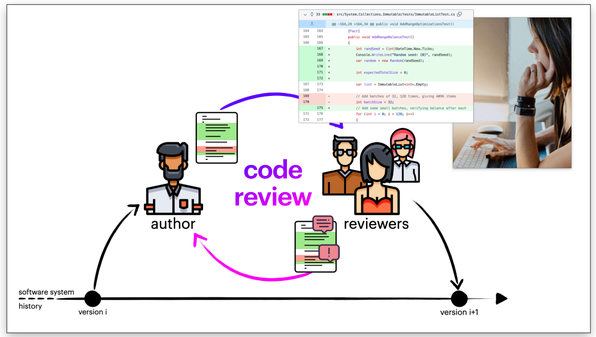


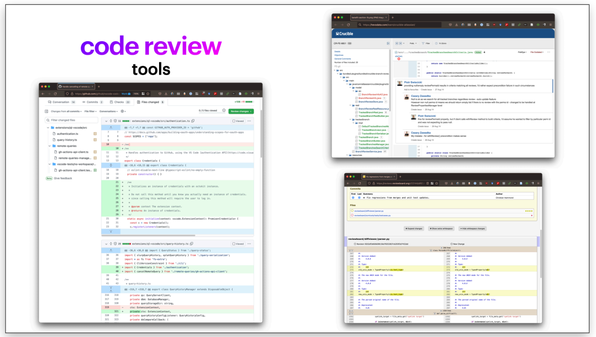


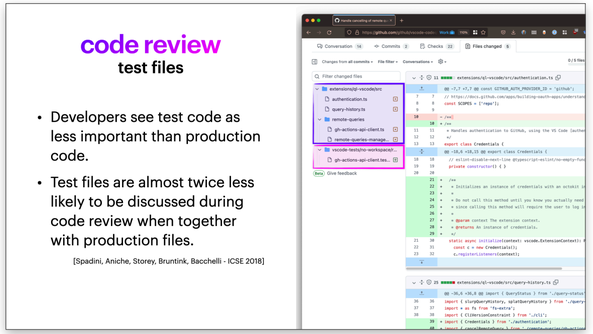
takeaways

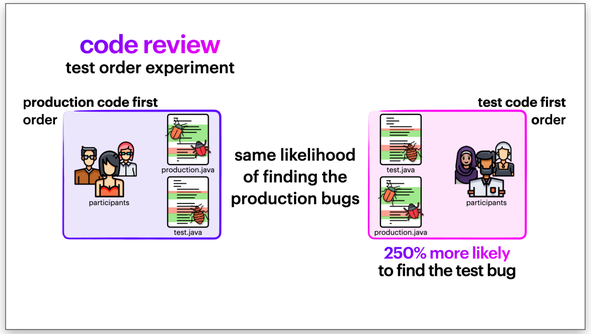
- **reviewers**: be aware of this effect and decide where to start your review in a principled way.
- **authors**: guide the reviewers towards the most challenging part of your change.
- **tool builders**: empower users to choose how to order their changes and more.

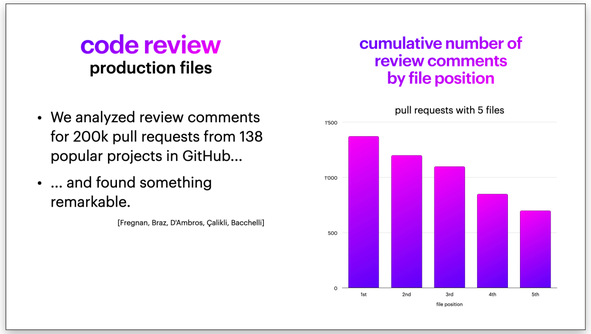
review

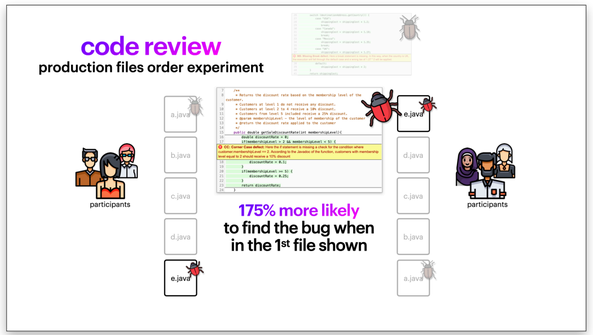








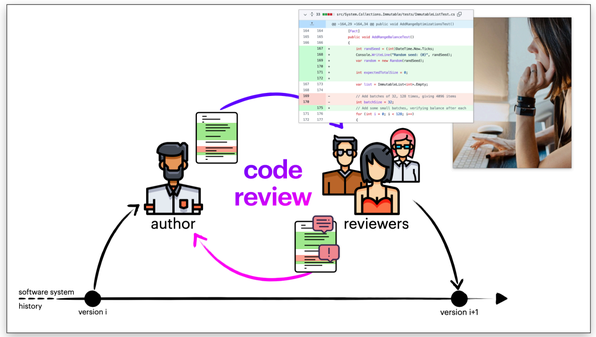


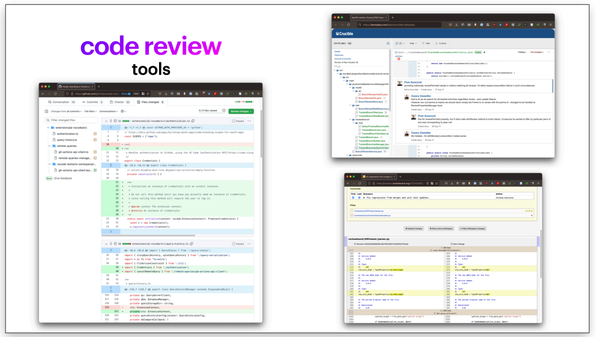


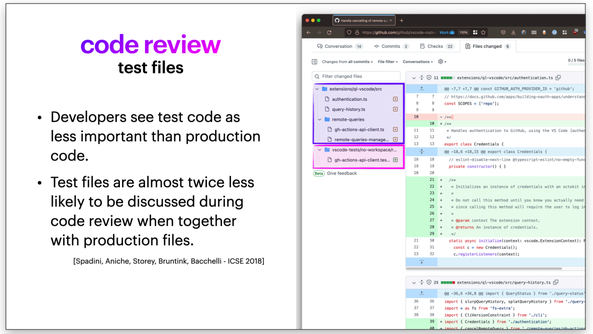
takeaways

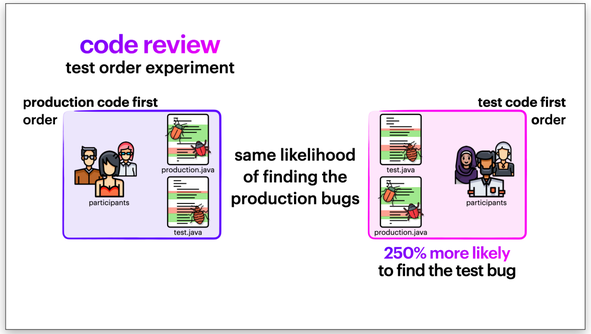
- **reviewers**: be aware of this effect and decide where to start your review in a principled way.
- **authors**: guide the reviewers towards the most challening part of your change.
- **tool builders**: empower users to choose how to order their changes and more.
- **everybody**: do not forget the power of the default settings.

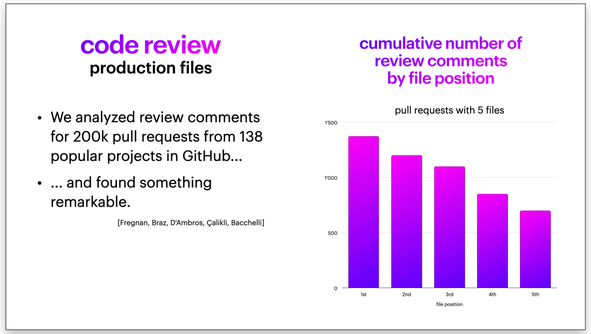
review

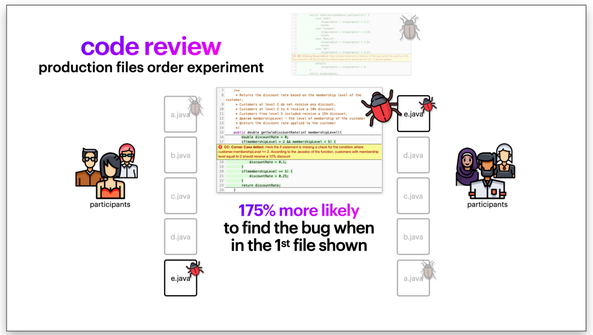












takeaways

- **reviewers**: be aware of this effect and decide where to start your review in a principled way.
- **authors**: guide the reviewers towards the most challening part of your change.
- **tool builders**: empower users to choose how to order their changes and more.
- **everybody**: do not forget the power of the default settings.

Credits: most presentation icons created by justicon & freepik @ Flaticon

Alberto Bacchelli
ASSOCIATE PROFESSOR
HEAD OF ZEST

zest
University of Zurich[UZH]

Binzmühlestrasse 14, 8050 Zurich, Switzerland
ADDRESS

zest@ifi.uzh.ch
EMAIL

@ZESTuzh
TWITTER

http://zest.ifi.uzh.ch
URL