Управління дефектами. Практика в JIRA

№ уроку: 13 **Курс:** Manual QA

Засоби навчання: Браузер, Jira-акаунт та середовище проекту Jira, Microsoft Office

Огляд, мета та призначення уроку

Мета даного уроку – вивчити, що таке дефект, з яких основних атрибутів повинен складатися звіт про дефекти, на практиці створити дефекти з різним контентом, а також навчити учнів використанню інструментів JIRA для ефективного управління дефектами в проектах розробки програмного забезпечення.

Вивчивши матеріал даного заняття, учень зможе:

- створювати та відстежувати дефекти, керувати процесом виправлення дефектів, аналізувати та вести звітність щодо процесу управління дефектами, а також щодо використання найкращих практик для ефективного управління дефектами;
- розібратися в основних поняттях та термінології управління дефектами, а також навчиться використанню процесів ведення дефектів.

Зміст уроку

1. Що таке баг? Причини виникнення дефектів.
2. Основні атрибути баг-репорту.
3. Життєвий цикл баг-репорту.
4. Що таке severity і priority.
5. Визначення дефекту. Навіщо та де описувати Bug report.
6. Аналіз, критерії, написання звітів про помилки.

Резюме

- **Керування дефектами** – це процес виявлення, відстеження та усунення помилок, які можуть виникнути під час розробки програмного забезпечення.
- **JIRA** – це інструмент для управління проектами розробки програмного забезпечення, який використовується для відстеження дефектів та управління задачами.
- **Дефект** – це будь-яка помилка, яка може виникнути у програмному забезпеченні. Дефект може проявлятися як у вигляді неправильної поведінки програми, неправильних результатів, некоректного інтерфейсу користувача тощо.
- **Процес управління дефектами** – це набір процедур та методів, які використовуються для виявлення, відстеження та усунення дефектів.
- **Життєвий цикл дефекту** – це послідовність етапів, які дефект проходить від виявлення до усунення. Життєвий цикл дефекту може включати такі етапи, як створення, призначення відповідального за виправлення, перевірка, закриття і т. д.
- **Класифікація дефектів** – це система класифікації помилок на основі їхньої пріоритетності та серйозності. Дефекти можуть бути класифіковані, наприклад, як критичні, високопріоритетні, середньої важливості, низької важливості і т. д.
- **Звітність про дефекти** – це процес аналізу та звітності про дефекти. Звітність може включати такі параметри, як кількість дефектів, статус дефектів, швидкість усунення дефектів тощо.
- Життєвий цикл дефектів

CyberBionic Systematics ®
19 Eugene Sverstyuk Str., 5 floor
Kyiv, Ukraine

Tel. 0 800 750 312
E-mail: edu@cbsystematics.com
Site: www.edu.cbsystematics.com | itvdn.com

Отже, ми знайшли баг. Може, навіть блокер. Що ж з ним може трапитись, на всьому його нелегкому життєвому шляху? (Назви етапів життя дефектів можуть бути різними в різних баг-трекінг системах, але суть їх одна).

- **Новий (New).** Тестувальник знайшов баг, дефект успішно занесений до «Bug-tracking» системи.
- **Відкритий (Opened).** Після того, як тестувальник відправив помилку, вона або автоматично, або вручну призначається на людину, яка повинна її проаналізувати (зазвичай Project Manager).
- Залежно від рішення менеджера проекту, баг може бути:
  - **Відкладений (Deferred).** Виправлення цього бага не несе цінності на даному етапі розробки або з інших причин, що відстрочують його виправлення.
  - **Відхилений (Rejected).** З різних причин дефект може і вважатися дефектом чи вважатися неактуальним дефектом, що змушує відхилити його.
  - **Дублікат (Duplicate).** Якщо описана помилка вже раніше була внесена до «Bug-tracking» системи, статус такої помилки змінюється на «дублікат».
  - **Призначений (Assigned).** Якщо помилка є актуальною і має бути виправлена в наступній збірці (build), відбувається призначення на розробника який має виправити помилку.
- Якщо наявність дефекту незаперечна, його шлях може призвести до таких статусів:
  - **Виправлений (Fixed).** Відповідальний за виправлення бага розробник заявляє, що усунув дефект.В зависимости от того, исправил ли разработчик дефект, дефект может быть:
  - **Перевірений (Verified).** Тестувальник перевіряє, чи справді відповідальний розробник виправив дефект, чи все-таки розробник безвідповідальний. Якщо бага більше немає, він набуває цього статусу.
  - **Повторно відкритий (Reopened).** Якщо побоювання тестувальника виправдані і баг у новому білді не виправлений – він так само вимагатиме виправлення, тому заново відкривається.
  - **Закритий (Closed).** Внаслідок певної кількості циклів баг все-таки остаточно усунений і більше не вимагатиме уваги команди – він оголошується закритим.

- **Класифікація дефектів**
  - Класифікація дефектів, з погляду ступеня впливу (Severity) на працездатність:
  - **Blocker**. Ошибка, которая приводит программу в нерабочее состояние. Дальнейшая работа с программной системой или ее функциями – невозможна.
  - **Critical**. Критический дефект, приводящий некоторый ключевой функционал в нерабочее состояние. Так же это может быть существенное отклонение от бизнес логики, неправильная реализация требуемых функций, потеря пользовательских данных и т.д.
  - **Major**. Весьма серьезная ошибка, свидетельствующая об отклонении от бизнес логики или нарушающая работу программы. Не имеет критического воздействия на приложение.
  - **Minor**. Незначительный дефект, не нарушающий функционал тестируемого приложения, но который является несоответствием ожидаемому результату. Например, ошибка дизайна.
  - **Trivial**. Баг, не имеющий влияние на функционал или работу программы, но который может быть обнаружен визуально. Например, ошибка в тексте.

- **Градація дефектів з точки зору пріоритетності виправлення (Priority):**

CyberBionic Systematics ®
19 Eugene Sverstyuk Str., 5 floor
Kyiv, Ukraine

Tel. 0 800 750 312
E-mail: edu@cbsystematics.com
Site: www.edu.cbsystematics.com | itvdn.com

- **High.** Баг може бути виправлений якнайшвидше, т.к. він критично впливає працездатність програми.
- **Medium.** Дефект повинен бути обов'язково виправлений, але він не чинить критичного впливу на роботу застосунку.
- **Low.** Помилка має бути виправлена, але вона не має критичного впливу на програму і усунення може бути відкладено, залежно від інших більш пріоритетних дефектів.

Закріплення матеріалу

1. Що таке управління дефектами і яке значення воно має в процесі розробки ПЗ?
2. Які етапи включає життєвий цикл дефекту у JIRA?
3. Які види класифікації дефектів існують і як вони використовують у процесі управління дефектами?
4. Як створити дефект у JIRA і які основні поля мають бути заповнені при створенні дефекту?
5. Як призначити відповідального за виправлення дефекту в JIRA і як відстежувати процес виправлення дефекту?

Самостійна діяльність учня

Завдання 1
Створіть нову задачу в JIRA, яка буде відноситися до управління дефектами. Дайте їй назву "Виправлення помилки з авторизацією". Заповніть поля "Опис", "Пріоритет" та "Відповідальний". Призначте себе відповідальним за виправлення помилки. Призначте високий пріоритет завданню, щоб воно було виконане якнайшвидше.

Завдання 2
Створіть звіт у JIRA про стан дефектів у вашому проекті. Відфільтруйте завдання за статусом "Відкрита" та "У процесі". Відобразіть у звіті кількість дефектів, час, витрачений на їх виправлення, та прогрес за кожним завданням. Поділіться звітом із вашою командою, щоб усі були в курсі стану дефектів та прогресу щодо їх виправлення.

Завдання 3
Проведіть ретроспективу щодо управління дефектами в минулому спринті. Створіть завдання у JIRA з назвою "Ретроспектива з управління дефектами в спринті N". У задачі опишіть, які дефекти були знайдені в минулому спринті, як вони були виправлені та як можна покращити процес керування дефектами у майбутньому. Дайте коментарі до кожного знайденого дефекту та запропонуйте ідеї для покращення процесу управління дефектами. Розподіліть завдання між учасниками команди, щоб кожен міг працювати над покращенням процесу.

Рекомендовані ресурси

1. Atlassian University – This website provides a variety of JIRA courses, including courses on Defect Management. The courses are self-paced and cover a range of topics, from the basics to advanced techniques. Website: https://www.atlassian.com/university

2. Agile Alliance – This website offers a range of resources on agile practices, including defect management. There are articles, videos, and webinars that provide insight into best practices for using JIRA for defect management in agile environments. Website: https://www.agilealliance.org/

CyberBionic Systematics ®
19 Eugene Sverstyuk Str., 5 floor
Kyiv, Ukraine

Tel. 0 800 750 312
E-mail: edu@cbsystematics.com
Site: www.edu.cbsystematics.com | itvdn.com

3. JIRA Tutorial – This website provides a comprehensive tutorial on JIRA, including defect management best practices. The tutorial includes videos and step-by-step instructions to help users get started with JIRA. Website: https://www.tutorialspoint.com/jira/

4. Agile Testing Days – This website provides a range of resources on agile testing practices, including defect management. There are articles, videos, and webinars that provide insight into best practices for using JIRA for defect management in agile environments. Website: https://agiletestingdays.com/

CyberBionic Systematics ®
19 Eugene Sverstyuk Str., 5 floor
Kyiv, Ukraine

Tel. 0 800 750 312
E-mail: edu@cbsystematics.com
Site: www.edu.cbsystematics.com | itvdn.com