|  |
| --- |
|  |
| **Travail pratique #4 – Les revues logicielles** |
|  |

Simon Delisle - 1538886

Félix Gingras Harvey - 1490242

François Pierre Doray - 1529403

Alexandre Vanier - 1525768

|  |
| --- |
|  |

# Article #1 - Expectations, Outcomes, and Challenges of Modern Code Review

## Chaîne de recherche

reviews AND (("IEEE Software" wn ST) OR ("Communications of the ACM" wn ST) OR ("Software Quality Journal" wn ST) OR ("Journal of Systems and Software" wn ST) OR ("ICSE" wn CF) OR ("ICSSP" wn CF))

## Justification du choix de l’article

L’article est intéressant, car il nous éclaire sur la revue de code dite moderne. En effet, nous avons choisi cet article pour mieux connaître les motivations, les résultats et les défis de la revue de code. De plus, les auteurs donnent des recommandations pour optimiser les revues qui pourront nous être utiles dans le futur.

## Description de l’article

Les méthodes de revues de code évoluent avec les années. Celles-ci deviennent moins formelles, plus “légères” et assistées par des outils, afin d’en réduire l'inefficacité. L’article cherche donc à fournir des données empiriques sur ces méthodes de revues de code utilisées aujourd’hui en entreprises. En effet, les auteurs ont récolté des données sur les motivations des revues de code, sur les résultats des revues de code et sur les plus grand défis lors de la revue de code. La récolte de données se fait en entreprise (Microsoft) avec les développeurs et les “manager”. L’outil utilisé par Microsoft pour faire des revues de code (CodeFlow).

L’analyse des revues concerne le code source et les auteurs ne mentionnent pas d’autre artefact. Par contre, comme le but de l’article est de cerner les différences entre les attentes envers les revues et les résultats obtenus, l’étude traite donc de la méthode elle-même et moins de l’artefact concerné, soit le code source.

## Associer la méthode de revue à son type

L’article traite de la revue du code source. Les revues de code serviront à voir si le code est conforme aux conventions de l’équipe, s’il ne contient pas de défauts. De plus, il contribue à l’amélioration du code et au transfert de connaissance. Il s’agit donc d’une inspection selon la norme IEEE 1028.

« Examples of software products subject to inspections include [...] Source code. »

«The purpose of an inspection is to detect and identify software product anomalies. An inspection is a systematic peer examination that does one or more of the following:

a) Verifies that the software product satisfies its specifications

b) Verifies that the software product exhibits specified quality attributes

c) Verifies that the software product conforms to applicable regulations, standards, guidelines, plans, specifications, and procedures. »

## Associer la méthode de revue à un ou plusieurs facteurs de qualité ISO 25000

### ISO 25010:

L’article concerne seulement le code source, donc nous ne pouvons pas faire de lien avec les facteurs de qualité à l’utilisation selon l’ISO 25010. Par contre, on peut associer la méthode de revue aux facteurs de qualité de produit, plus précisément le «Functional suitability» (4.2.1). En effet, les revues permettent de détecter les défauts, d’améliorer le code et d’assurer que le code fonctionnera correctement.

*«Almost all the managers included finding defects as one of the reasons for doing code reviews»*

*«Programmers ranked code improvement as an important motivation for code review, close to finding defects»*

### ISO 25012:

L’article comporte beaucoup de données telles que des statistiques sur la motivation des revues de code, les commentaires dans l’outil de revue de code et le code source. Elles ne sont pas reliées directement à l’activité de revue. Par contre, elles pourront aider les développeurs à faire de meilleures revues de code.

En ce qui concerne les associations avec la norme, nous avons pu identifier l’«accessibility», la «traceability», l’«accuracy» et l’«understandability». En ce qui concerne l’«accessibility», les développeurs ont accès à un outil en ligne qui leur permet d’avoir accès aux commentaires de manière rapide et en tout temps. Pour la «traceability», l’outil permet d’associer les commentaires à une ligne de code spécifique. Il est donc facile de retracer les commentaires. Les dates sur les commentaires peuvent servir à les associer à des versions. L’«accuracy» permet d’avoir des résultats fidèles grâce aux candidats choisis aléatoirement. Bref la méthodologie est bonne. Puis l’«understandability» n’est pas respecté, car deux développeurs peuvent avoir des conversations informel sur le sujet, donc les connaissent ne sont pas partagé avec les autres développeurs.

De plus, toutes les données peuvent être utilisées pour analyser le processus d’une entreprise afin de pouvoir l’améliorer.

# Article #2 - An Empirical Analysis of Team Review Approaches for Teaching Quality Software Development

## Chaîne de recherche

technical reviews AND (("IEEE Software" wn ST) OR ("Communications of the ACM" wn ST) OR ("Software Quality Journal" wn ST) OR ("Journal of Systems and Software" wn ST) OR ("ICSE" wn CF) OR ("ICSSP" wn CF))

## Justification du choix de l’article

Cet article est intéressant par le fait qu’il compare quelques techniques de revues entre elles, et démontre exhaustivement comment quelques variations à la technique de revue utilisée peuvent faire en faire varier l’efficacité.

## Description de l’article

Une étude comparative entre plusieurs variations de types de revue est faite et les résultats y sont présentés. L’efficacité de chaque variation est présentée sous forme de taux de découverte de défauts de chaque sorte. Les revues furent effectuées sur les SRS, les diagrammes de classes et les diagrammes de séquences créés par les étudiants.

L’article traite donc de plusieurs facteurs de qualité pour ces artefacts, soit l’organisation, la complétion, la justesse, la cohérence, la précision et la traçabilité (pour la spécification des requis) ainsi que la complétion, l’utilisation de l’héritage et de l'agrégation, la cohésion, la traçabilité, et le couplage (pour le design architectural).

*“The checklist focused on the basic quality attributes of requirements and the SRS document: organization, completeness, correctness, consistency, lack of ambiguity, non-functional aspects and traceability.”*

*“The checklist included questions related to completeness, organization, correctness, consistency, lack of ambiguity, modifiability and traceability.”*

Par contre les conclusions présentées dans l’étude sont centrées sur un facteur de qualité concernant les revues en tant que tel, soit l’efficacité, qui elle aura une influence sur les artefacts par la suite.

*“Our main findings are that CDR without author is better that CDR with author in terms of major defect identification in software requirements and design at the undergraduate level. Secondly, SDR is more effective in defect detection process than CDR.”*

## Associer la méthode de revue à son type

L’article traite des revues techniques au niveau de la spécification des requis et de l’analyse et du design. En effet, les étudiants participant à l’étude évaluaient des artefacts créés par des équipes semblables par rapport à des grilles d’évaluations tirées de standards établis.

*“A tailored version of the defect checklist [9, 18] was developed. Questions were simplified and advanced concepts were removed to increase understandability and performance. The checklist focused on the basic quality attributes of requirements and the SRS document: organization, completeness, correctness, consistency, lack of ambiguity, non-functional aspects and traceability.”*

Cela rejoint donc la définition de l’IEEE 1028 sur les revues techniques : *“The purpose of a technical review is to evaluate a software product by a team of qualified personnel to  determine its suitability for its intended use and identify discrepancies from specifications and standards.”*

À l’exception près que le personnel peut difficilement être considéré comme étant qualifié puisque les étudiants sont en formation et donc en voie d’être qualifiés.

## Associer la méthode de revue à un ou plusieurs facteurs de qualité ISO 25000

### ISO 25010:

Comme l’étude traite de revues durant les phases de spécification des requis, cela aura un impact sur l’utilisation future du logiciel puisque l’on s’assure que les requis sont bien définis. Cela touche donc le facteur de qualité dans l’utilisation selon l’ISO 25010, soit la satisfaction (4.1.3).

De plus, comme l’étude traite également des revues durant l’analyse et le design architectural, cela aura un impact sur la transitions des requis en fonctionnalités et touche donc le niveau produit selon l’ISO 25010, et plus précisément la pertinence fonctionnelle (4.2.1).

### ISO 25012:

Comme dit précédemment, plusieurs facteurs de qualité étaient évalués durant l'étude sur l’information contenue dans les différents artefacts. Plusieurs d’entre-eux sont décrits dans l’ISO 25012, soit :

La cohérence ( “*Consistency”*)

La complétion (“*Completeness”*)

La justesse (“*Accuracy”*)

La précision (“*Precision”*)

La traçabilité (“*Traceability”*)

Par contre, très peu d’information est donnée sur les données créées par les revues. Les étudiants devaient remplir une liste de cases à cocher selon des questions concernant les aspects de qualité étudiées. Il semblerait donc que les données étaient complètes et précises pour le cadre de l’étude, et permettaient donc de tester les différentes hypothèses.

# Conclusions sur les deux articles étudiés

En conclusion, nous avons déterminé que les deux articles étudiés touchent à un nombre relativement important de facteurs de qualité de la norme ISO25000. En effet, les deux articles sont associés à plus de 10 facteurs de qualité (principalement dans la norme ISO25012). Il est important de noter que ces associations n’ont pas toujours la même force. La rédaction de ce travail nous a permis de réaliser l’importance des revues, principalement les revues techniques. Cela nous a aussi permis de voir à quel point celles-ci sont intimement reliées à la qualité du logiciel.

L'article «Expectations, Outcomes, and Challenges of Modern Code Review» est particulièrement intéressant car il analyse l'application de la revue dans un contexte réel. Il présente les avantages d'outils auxquels nous avons accès et que nous pourrions utiliser dans notre vie professionnelle ou scolaire. Il trace un portrait concret des bénéfices que les gens souhaitent et réussissent à tirer de la revue, ce qui nous permettra de faire de meilleurs choix. Dans notre projet intégrateur, nous avons utilisé les outils de révision de code fournis par GitHub (similaire à CodeFlow, mais allégé). Les résultats de cet article nous ont permis de réfléchir sur les bénéfices que nous pouvons en retirer, ce que nous n'avions pas réellement fait auparavant. Il serait intéressant d’utiliser CodeFlow et refaire des inspections de parties critiques de notre projet et en analyser les conséquences.

Alors que le premier article nous a permis de constater les bénéfices de la revue, l'article «An Empirical Analysis of Team Review Approaches for Teaching Quality Software Development» se concentre sur le déroulement de celle-ci. Il nous sera utile car il présente les pratiques que nous pouvons faire varier pour optimiser l'efficacité du processus de revue. Entre autres, il est dit clairement qu'un cross-domain review (CDR) est plus efficace lorsque l'auteur est absent. Dans le contexte de notre projet, cela est sans doute plus dure à mettre en œuvre puisque la portée et limité et nous travaillons tous dans le cadre d’un même domaine qui englobe le projet.