STONE: A Software Configuration Management Process for Academic Projects

Rebeca M. Pontes[1] and Valéria Lelli[2]

[1] Universidade Federal do Ceará (UFC), Campus Russas, Russas, Brasil
rebeca.maia@alu.ufc.br
[2] Universidade Federal do Ceará (UFC), Fortaleza, Brasil
Grupo de Redes de Computadores, Engenharia de Software e Sistemas (GREat)
valerialelli@ufc.br

Abstract. The Software Configuration Management (SCM) is essential to control the changes that occur within a project. Although its application has many benefits, its use isn't encouraged in projects developed by students during their undergraduate course. With a lack of a SCM process that meets the student's needs, their projects do not follow rules to incorporate changes and their artifacts could become inconsistent. In this context, this paper proposes the STONE process for SCM of academic projects. To define the process, a survey was conducted with 67 undergraduate students from the Computer Science course and related areas, through a questionnaire to identify how students dealt with SCM within the projects in which they have participated. The process was applied to an undergraduate class, where the process flow of activities was exemplified using a game. The results showed that the process helped students to manage the changes in their academic projects and the templates provided to generate the process artifacts were essentials to facilitate the execution of the process.

Keywords: Configuration Management, Academic Projects, Software Process.

1 Introdução

A Gerência de Configuração de Software (GCS) desempenha um papel fundamental no processo de desenvolvimento, pois ela se preocupa em realizar todas as atividades que envolvem o controle de mudanças que ocorrem dentro de um projeto de software. Na GCS mantém-se não somente o artefato, mas o que foi modificado, quando e como foi modificado, quem modificou, quanto tempo levou a modificação, qual o impacto da mudança, dentre outras informações [13].

O ensino de Engenharia de Software (ES) e de cursos de áreas afins da Computação estabelece que os alunos devem ter alguma experiência prática nas disciplinas que cursam. Para que os alunos adquiram essa experiência, eles geralmente desenvolvem sistemas ao longo das disciplinas do curso com o intuito de consolidar o conhecimento acerca de determinado tópico [1]. Nesses projetos, à medida que os artefatos de software são desenvolvidos, mudanças naturalmente ocorrem seja pelo amadurecimento do conhecimento do aluno ou por solicitação do professor/tutor. O uso de GCS nesses projetos é feito de maneira bem informal e

limitada, pois quando a disciplina de GCS faz parte da matriz curricular do curso, ela é optativa e/ou ofertada nos semestres mais avançados do curso. Dessa forma, quando aquelas mudanças que ocorrem não são gerenciadas corretamente pelos alunos, acarretando em vários problemas, por exemplo, os artefatos são entregues fora do prazo estabelecido pelo professor ou não atendem os requisitos ou falta consistência entre eles. Esses problemas evidenciam a dificuldade dos alunos em lidar com a capacidade de mutabilidade inerente a todos os projetos de desenvolvimento de software, uma característica essencial entre os profissionais da área de Tecnologia da Informação.

Para lidar com os problemas mencionados, alguns estudos de GCS com foco em projetos acadêmicos [2][5][12][16] se preocupam apenas com a fase de codificação e controle de versão do código, negligenciando as outras atividades desenvolvidas pelos alunos, como a documentação dos requisitos do sistema, a modelagem do projeto como a elaboração de diagramas, dentre outras.

Até o presente momento, não foram identificados na literatura estudos que abordassem as principais atividades de GCS - como 'Identificação dos Itens de Configuração', 'Planejamento da Configuração', 'Controle de Mudança' - para auxiliar sua aderência nos projetos desenvolvidos pelos alunos durante seu período na universidade. Portanto, o objetivo deste trabalho é propor um processo de GCS voltado especificamente para projetos acadêmicos com o intuito de auxiliar os alunos a gerenciar as mudanças dos artefatos produzidos nas disciplinas, podendo utilizá-lo nos semestres iniciais do curso, além de familiarizá-los com as técnicas e procedimentos de GCS.

Este trabalho está organizado da seguinte forma: na Seção 2, os principais conceitos abordados na pesquisa são apresentados. Na Seção 3 é apresentada a metodologia utilizada para a definição do processo proposto. Na Seção 4, os resultados do questionário são apresentados. Na Seção 5 é feita a descrição do processo de GCS proposto. Na Seção 6 é apresentado os resultados da aplicação do processo STONE. Na Seção 7 é descrito alguns dos trabalhos relacionados e, por fim, na Seção 8, são apresentadas as considerações finais deste trabalho.

2 Fundamentação Teórica

A GCS é uma subárea da Engenharia de Software que se refere ao processo pelo qual todos os artefatos relevantes para o projeto e os seus relacionamentos são armazenados, recuperados, identificados exclusivamente e modificados [7]. Cada artefato que precisa ser mantido sob o controle de mudanças é denominado um item de configuração (IC) [13]. Dessa forma, conforme os itens de configuração são produzidos, é necessário mantê-los em um repositório. O repositório é uma coleção de todos o os artefatos de software pertencentes a um sistema e a localização/formato na qual tal coleção é armazenada [10]. O propósito deste armazenamento é garantir que o IC não desaparecerá ou será danificado, de maneira que este possa ser encontrado a qualquer momento e entregue nas condições que se deseja encontrar.

A cada alteração que um IC for submetido, é preciso criar uma nova versão para este IC. A versão de um item de software é uma instância identificada de um item.

Pode ser visto como o estado de um item em evolução [8]. Comumente, um projeto de software possui inúmeros ICs e cada um desses precisam ter suas versões controladas.

Uma das tarefas que compõem a elaboração do PGC, é a definição da baseline. No contexto de ES, definimos uma baseline como um conjunto de itens de configuração que devem ser entregues em um determinado momento do projeto.

Quando um IC é liberado ao cliente, ele passa a ser denominado de *release*. Segundo a ISO/IEC 24765 [10], um *release* "é uma versão entregável de uma aplicação que pode incluir toda ou parte de uma aplicação, uma coleção de ICs novos e/ou alterados que foram testados e introduzidos em um ambiente". Dependendo do projeto, essas entregas podem ser feitas feitas incrementalmente. Quando um ou mais ICs são entregues pode-se definir um *baseline* como um marco de referência do desenvolvimento do software.

Todas as atividades de gerenciamento de configuração são guiadas pelo Plano de GCS que é um documento que descreve a abordagem de GCS usada bem como os procedimentos, políticas, responsabilidades, dentre outros.

3 Metodologia

Para a concepção do processo, inicialmente foi feita uma pesquisa bibliográfica para identificar os trabalhos existentes na literatura da área de GCS. Para essa pesquisa foi definida a seguinte string de busca:

(("configuration management" OR "change management" OR "version control" OR "versioning management" OR "version management" OR "code version" OR "release management") AND (process OR approach OR method OR technique OR tool OR pattern) AND ("small project" OR "student project" OR capstone OR short OR academic OR academia) AND ("software development" OR "application development" OR "system development"))

Vale ressaltar que para essa pesquisa não foi feito um mapeamento ou revisão sistemática. Dessa forma, a string de busca foi aplicada apenas em duas bases acadêmicas: ACM e IEEE. Para filtrar os artigos relevantes ao tema, foi feita a leitura dos seus resumos, e aqueles mais relacionados ao tema foram lidos por completo e catalogados através de um fichamento. Esse fichamento registrou o objetivo de cada trabalho e quais informações importantes poderiam ser incorporadas no processo. Na base da IEEE, 307 artigos foram retornados, dos quais 14 foram selecionados. Na ACM, 133 estudos foram extraídos, sendo que 11 foram selecionados, totalizando 25 artigos para esta pesquisa.

Em paralelo, também foi conduzido um estudo dos processos propostos por modelos de qualidade e normas técnicas que tratam GCS, tais como o SWEBOK [8], o Guia de Implementação do MPS-BR [15] e as normas técnicas IEEE 1042 [9] e ISO 10007 [3].

A partir dos dados levantados da pesquisa da literatura e do estudo dos modelos de qualidade, foi proposta a primeira versão do processo de GCS. Em seguida, foi realizada uma pesquisa de opinião através de um questionário para identificar as necessidades dos alunos em relação a GCS em projetos acadêmicos. Com base nos

resultados obtidos, foram definidos templates para facilitar a elaboração dos artefatos de GCS. Além disso, um jogo foi criado para que os alunos pudessem compreender melhor os conceitos de GCS e as atividades do processo previamente definido.

O processo foi utilizado em uma disciplina de graduação e seu uso foi avaliado através de um questionário aplicado aos alunos e dos artefatos de GCS que eles produziram. Também foram avaliadas duas ferramentas para o "Controle de Mudanças" sugeridas no processo.

4 Questionário

A pesquisa foi realizada através da aplicação de um questionário online[1] em turmas do curso de Computação e cursos afins da Universidade Federal do Ceará (UFC), obtendo-se um total de 67 participantes. O objetivo do questionário é identificar como os alunos lidavam com a GCS dentro dos projetos em que eles participavam, bem como ressaltar as necessidades que os alunos tinham em relação a GCS.

Sobre o perfil dos alunos, 34 são do curso de Engenharia de Software (ES), seguido pelos cursos de Ciência da Computação (CC) (24), Engenharia da Computação (EngComp) (8) e Sistemas e Mídias Digitais (SMD) (1). A maioria dos alunos (61) já cursaram mais da metade do curso.

Em relação ao conhecimento de GCS, 66 alunos afirmaram que já cursaram uma ou mais disciplinas que introduziram conceitos de GCS. Conforme é ilustrado na Fig. 1, a mais citadas forma Engenharia de Software (32 alunos), Processos de Software (27 alunos) e Qualidade de Software (24 alunos) para os alunos do curso de ES, e a disciplina de Engenharia de Software para os participantes do curso de CC, EngComp e SMD com 25, 7 e 1 respondentes, respectivamente. Na matriz curricular desses cursos, observa-se que essas disciplinas são ofertadas a partir do 5º semestre. Porém, teve 1 aluno do CC que apesar de estar no 6º semestre respondeu que não tinha cursado nenhuma disciplina que abordasse GCS.

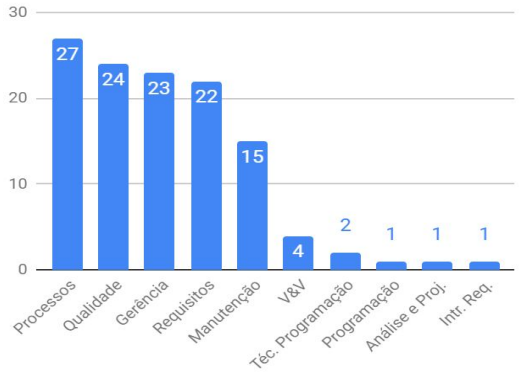

Fig. 1. Disciplinas que abordam conceitos de GCS.

[1] https://forms.gle/yN6DvCsZR5YX1XDK6

Dentre os produtos de trabalho mais produzidos pelos participantes nas disciplinas são: documento de requisitos (52 alunos), cronograma de atividades de projeto (42 alunos), descrição dos papéis e responsabilidades da equipe (41 alunos), código-fonte (34 alunos), plano arquitetural (21 alunos) e plano de teste (4 alunos). Outros artefatos como Termo de Abertura do Projeto (2 alunos), Plano de Projeto (2 alunos) e Relatório de Viabilidade (1 aluno) são produzidos em menor escala. Embora alguns dos produtos de trabalho mencionados, como o cronograma de atividades e a descrição das responsabilidades da equipe, não sejam comumente elaborados separadamente de outros artefatos de ES, observa-se que os alunos costumam elaborá-los para planejar a execução das atividades de seus projetos. Também, observou-se que a maioria desses produtos são entregues ao professor pelos alunos de forma incremental (42), enquanto em outros projetos de disciplinas é realizada apenas uma entrega (25 alunos). Vale ressaltar que nas entregas incrementais a quantidade de artefatos produzidos e entregues tendem a ser maior, o que implica em mais releases, mais relatórios de solicitação de mudança, mais alterações no Plano de Gerência de Configuração, dentre outros. Por outro lado, quando o projeto se baseia em uma única entrega, os alunos costumam fazer as atividades de GCS só no final do projeto, como versionamento e armazenamento dos artefatos.

No que diz respeito ao controle de mudanças, 44 alunos apontaram que a "Política de controle de versões" é essencial para ajudar no controle de mudanças dos artefatos produzidos nas disciplinas, seguido pela "Definição de ferramentas para controle de mudanças" (32), pela "Descrição das solicitações de mudança" (23), pelos "Metadados dos artefatos produzidos" (16) e, pela "Distinção da urgência da mudança" (9). Embora a maioria tenha mencionado a política de controle de versões como principal informação para a aplicação da GCS, a frequência com que eles realizam as atividades de controle de versão ainda deixa a desejar. Embora 32 alunos façam corretamente o controle de versão, *i.e*., a cada alteração feita no artefato, 18 alunos apenas fazem o controle nas versões entregue ao professor, 7 alunos fazem o controle a cada marco no projeto, 7 não fazem controle de versão, 1 faz versionamento somente em código-fonte e 1 faz controle de versão de acordo com o decorrer do desenvolvimento. Isso indica que a maioria (34) dos alunos têm negligenciado essa atividade fundamental para o processo de GCS, ocasionando problemas na manutenção das versões corretas dos artefatos.

Uma das perguntas aborda sobre quem geralmente valida as alterações nos artefatos, e grande parte (50) são os próprios alunos, membros de equipe, sendo que o restante (17 alunos) respondeu que quem valida é o próprio professor. Esse resultado aponta que os alunos fazem várias alterações antes de entregar o artefato requerido ao professor. Nesse caso, é importante que os alunos saibam gerenciar as alterações para que não ocasione inconsistências nos artefatos.

No que diz respeito ao planejamento das atividades, 53 alunos realizam um planejamento prévio, enquanto o restante (14 alunos) não realiza. Dentre as ferramentas mencionadas, a mais utilizada é o Trello (42) e o Kanban (5) conforme ilustra a Fig. 2. Por outro lado, em relação à ferramenta de controle de versão 51 alunos responderam que utilizam o Git, e apenas um utiliza o mecanismo do Google Drive para manter as versões dos artefatos (e.g., histórico de versões do Google Docs). 15 alunos responderam que não utilizam nenhuma ferramenta para

planejamento das atividades e, similarmente, 15 não utilizam ferramentas para o controle de versões.

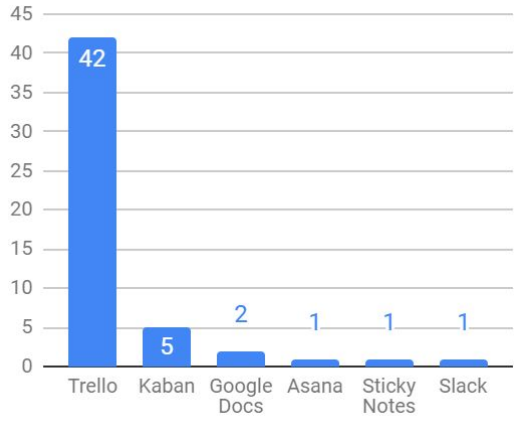

Fig. 2. Ferramentas utilizadas no Planejamento.

Sobre o histórico de alterações, 38 alunos costumam utilizar dentro dos seus documentos (e.g., histórico de revisões), mas o restante (29 alunos) não utiliza. Dos 38 alunos que usam o histórico de alterações, 35 apontam que é essencial incluir quem alterou o artefato, seguida pela data de modificação (35 alunos), o que foi alterado (32 alunos), o motivo da mudança (23 alunos) e o número da versão (21 alunos). O histórico de alterações é útil principalmente para o aluno que não utiliza nenhuma ferramenta para controle de versões e tem o propósito de apontar o que mudou no artefato, quem mudou, a data da modificação e a última versão do artefato.

Em relação ao problemas enfrentados pelos alunos devido à inconsistência entre os artefatos e suas versões, as causas mais apontadas foram: falta de manutenção no histórico de versões (17); alteração de artefato sem o conhecimento da equipe (11); não utilização de ferramentas para controle de mudanças (9); não utilização de um repositório para manter os artefatos (8); entrega de uma versão incorreta de artefato (7); e perda de versões de artefato (3 alunos). Para solucionar os problemas mencionados 30 alunos corrigiram a inconsistência manualmente, 3 alunos optaram por ignorar o problema e 2 fizeram uso de uma ferramenta para corrigir a inconsistência.

Sobre o uso de repositórios para manter os artefatos produzidos, 61 alunos afirmam que utilizam, mas 6 não utilizam. Conforme ilustra a Fig. 3, 45 utilizam o repositório do Google Drive. Outras ferramentas apontadas foram o Github (35), que é voltado para código-fonte, Dropbox (9), Bitbucket (1), Overleaf (1) e Gitlab (1). Dos 6 alunos que não utilizam repositório, todos afirmam que usam o computador pessoal como repositório.

Com o intuito de saber a opinião dos alunos se é melhor delegar a responsabilidade da GCS a uma única pessoa ou à equipe toda, 52 alunos concordaram que é melhor ter um membro do projeto que seja responsável pela aplicação do processo de GCS por motivos de organização. Com uma pessoa coordenando fica mais fácil de executar as atividades previstas. Porém, 11 alunos

afirmam que todos da equipe devem ser responsáveis pela execução do processo de GCS e não tem necessidade de apenas uma pessoa assumir toda a responsabilidade.

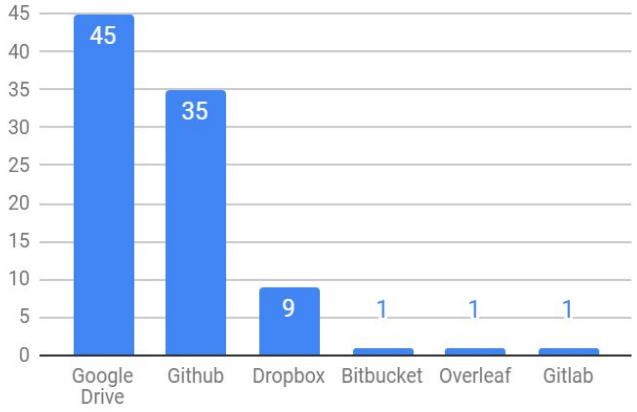

Fig. 3. Repositórios utilizados pelos alunos.

5 Processo STONE

Nesta seção é apresentado o processo STONE (SofTware cOnfiguration maNagemEnt for academic projects) e como este foi definido.

5.1 Concepção do Processo

O processo proposto foi embasado em normas técnicas [3][9], modelos de qualidade [15] e boas práticas de Engenharia de Software [7] que tratam de GCS. A norma técnica IEEE 1042 [9], que fornece orientação às práticas de planejamento de GCS, serviu como base para a definição das fases do processo STONE. Por outro lado, a norma ISO 10007 [3] que provê diretrizes de GCS com o intuito de prover visibilidade e controle das características funcionais do produto de software serviu para definir algumas atividades do processo. Por exemplo, a atividade "Identificar a Configuração" da fase de "Planejamento" foi adaptada dessa norma para se adequar ao escopo do processo, levando-se em consideração o tamanho, a complexidade e a natureza do trabalho.

O SWEBOK [8] serviu como base para a definição das atividades de planejamento da GCS, pois este guia apresenta uma visão de GCS que se relaciona com Garantia de Qualidade de Software (SQA). No SWEBOK, a fase de "Gerenciamento do processo de GCS" contém a atividade de "Planejamento da GCS", que no processo proposto foi adaptada para a atividade "Elaborar Plano de GCS" da fase de Planejamento.

Para a definição das atividades da fase de "Controle de Mudanças" utilizou-se como base o resultado esperado "Modificações em itens de configuração são controladas" da área de conhecimento de GCS do Guia de Implementação do MPS-BR [15]. Já as atividades das fases de "Controle de Versão" e "Gerenciamento

de *Releases*" do processo STONE se baseiam em atividades propostas pelo modelo de processo Rugby, proposto por Krusche et al. [11].

Além das normas e modelos supracitados, algumas atividades foram definidas para se adequar ao processo, tais como "Preparar o Ambiente de Configuração" e "Armazenar itens de configuração produzidos".

5.2 **Descrição das atividades do Processo STONE**

O processo, ilustrado na Fig. 4, é dividido em 4 fases: Planejamento; Controle de Mudanças; Controle de Versão; e Gerenciamento de Releases. Para cada fase foi definido um conjunto de atividades, que por sua vez, possui tarefas a serem realizadas. As atividades têm como saída um novo artefato ou a atualização de um artefato já produzido anteriormente em alguma atividade. Na Figura 4 é apresentada uma visão geral do processo com as suas atividades organizadas de acordo com as fases, o fluxo dessas atividades e os seus artefatos de entrada e saída.

A fase inicial do processo STONE é o "Planejamento". Nessa fase os artefatos do projeto que devem ficar sob o controle de mudanças são identificados e são definidas atividades para elaboração do Plano de Gerência de Configuração (PGC), o principal artefato do processo. As seguintes atividades fazem parte dessa fase:

1. Identificar a configuração: os itens de configuração (ICs) são selecionados e documentados. À medida que novos artefatos são produzidos durante o projeto, é necessário atualizar o PGC.
2. Preparar ambiente de configuração: é definido o ambiente de configuração necessário para a implementação do processo de acordo com os ICs identificados no PGC.
3. Elaborar Plano de Gerência de Configuração: o plano contém a descrição dos papéis e responsabilidades dos membros do projeto, as ferramentas selecionadas para apoiar a execução do processo, a rastreabilidade entre os ICs e a definição dos marcos do projeto.

Na fase de "Controle de Mudanças", os ICs que sofrerem alterações passarão por uma sequência de atividades a fim de controlar as mudanças e manter a consistência entre os demais artefatos do projeto. Ao contrário das outras fases, esta poderá ser iniciada a qualquer momento durante o projeto desde que as atividades da fase de Planejamento tenham sido executadas. As atividades dessa fase são:

1. Solicitar mudança: para cada IC que precisa de alteração, deve ser feito o registro da solicitação através do formulário de solicitação de mudança.
2. Analisar impacto da mudança: após o registro da solicitação da mudança ter sido feito e os demais membros da equipe terem sido notificados sobre esta solicitação, é necessário analisar o impacto da mudança nos ICs que serão afetados.
3. Implementar mudança: caso seja aprovada a implementação da mudança, o IC que será alterado é encaminhado a um membro da equipe que tem conhecimentos para implementá-la e deve ser gerado um relatório de mudança.

Na fase de “Controle de Versão”, os ICs que foram alterados na fase de “Controle de Mudanças” devem ter suas versões atualizadas e, caso se aplique, estas devem ser validadas para verificar se a mudança no IC foi implementada corretamente. As seguintes atividades fazem parte dessa fase:

1. Versionar item de configuração que sofreu mudança: após alterado, o IC deve ter sua versão atualizada de acordo com as regras definidas no PGC.
2. Validar IC alterado (optativa): após a alteração, pode ser necessário validar o IC. Porém, essa atividade é optativa e para ser executada dependerá do escopo do projeto.

Na fase de “Gerenciamento de Releases” os ICs que tiveram alguma mudança são entregues ao solicitante, no caso, das disciplinas, o professor ou tutor. As seguintes atividades são executadas:

1. Entregar item de configuração: se o IC precisar ser formalmente entregue ao solicitante, este será encaminhado pelo canal de comunicação definido no PGC.
2. Registrar feedback: após o IC ter sido entregue, o feedback do solicitante deverá ser registrado e armazenado. Se necessitar de uma nova alteração, esta deverá ser encaminhada para a fase de “Controle de Mudanças”.

Ao final de cada fase, na atividade “Armazenar itens de configuração produzidos”, os ICs que são alterados e os artefatos gerados são armazenados. Quando todos os artefatos do projeto são entregues ao professor, e nenhuma alteração é requerida, o processo é finalizado. Os templates dos artefatos de GCS são fornecidos aos alunos, eles foram elaborados a partir de outros trabalhos na literatura [9] [3] e adaptados ao contexto dos projetos acadêmicos.

Para a execução das atividades apresentadas do processo STONE foi definido dois papéis: aluno e professor e/ou tutor. Cada papel corresponde a um papel funcional no projeto. Por exemplo, o professor ou tutor assume o papel funcional de “Cliente”, o qual solicita o software a ser desenvolvido e também pode demandar qualquer alteração nos ICs. Por outro lado, as equipes, que são formadas pelos os alunos, podem assumir diferentes papéis funcionais que podem ser engenheiros de requisitos, desenvolvedores, dentre outros. A escolha dos papéis funcionais dependerá do escopo da disciplina do curso que pode ser voltado para o desenvolvimento do software como um todo (e.g., trabalhos da disciplina de ES) ou apenas focado em uma determinada fase do ciclo de vida do software (e.g., trabalhos da disciplina Verificação e Validação de Software). Tanto o aluno quanto o professor podem solicitar mudanças nos itens de configuração desenvolvidos ao longo do projeto.

Para viabilizar a produção e armazenamentos dos itens de configuração e facilitar a comunicação entre os membros do projeto algumas ferramentas são sugeridas no processo. Por exemplo, o editor de planilhas e de textos (e.g., Google Docs e Sheets, Word) podem ser utilizados para elaborar o formulário de solicitação de mudança, o relatório de mudanças, o PGC, dentre outros.

Para armazenar os itens e artefatos de configuração são sugeridos repositórios, tais como Google Drive, DropBox, dentre outros. Para o controle de mudanças e de versões dos ICs são sugeridos softwares como Mantis para o planejamento de atividades e o Git para versionamento de código.

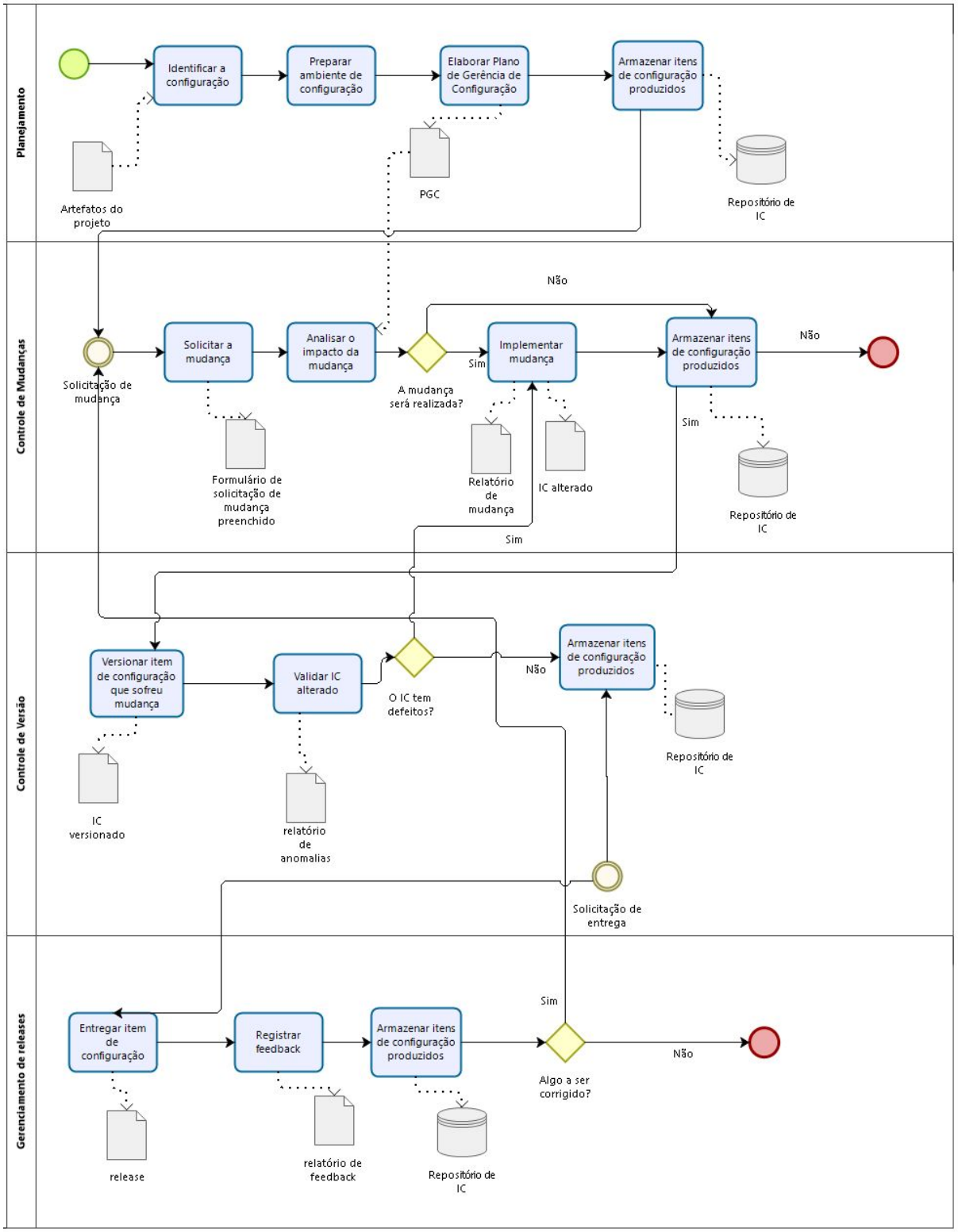

Fig. 4. Processo STONE

6 Aplicação do Processo STONE

Com base nos resultados obtidos no questionário, constatou-se a necessidade de elaborar um jogo de cartas para que os alunos pudessem compreender melhor os conceitos de GCS e as atividades do processo STONE. Dessa forma, o jogo foi

aplicado na disciplina optativa de Verificação, Validação e Teste de Software, ofertada para os cursos de CC, EngComp, dentre outros da Universidade Federal do Ceará. A dinâmica do jogo inicialmente consistiu na escolha dos alunos para assumir os papéis do processo STONE. O aluno que assumiu o papel de Gerente de Configuração sorteou uma carta de projeto e a partir dela foram selecionados os artefatos de entrada do processo. O jogo foi dividido pelas fases do processo. Em cada fase um jogador executa as atividades de acordo com o processo, selecionando os artefatos de entrada e saída e as ferramentas necessárias para cada atividade. Ao final da fase de Planejamento, é sorteada uma carta de evento que corresponde a uma solicitação de mudança no projeto. A partir dessa solicitação de mudança, os artefatos de controle de mudanças foram posicionados pelos jogadores, repetindo esse fluxo até a última fase Gerenciamento de Releases. No final do jogo, ganha o jogador que tiver a maior taxa de acerto no posicionamento correto dos artefatos de entrada e saída do processo.

Após o jogo, os alunos, em equipes, utilizaram o processo em artefatos da disciplina que sofreram alterações e, com base nesses artefatos, foram feitas as modificações usando o processo proposto. Os alunos produziram os artefatos iniciais do processo que foram o Plano de GCS, o Formulário de Solicitação de Mudança e o Relatório de Mudanças. Para cada um desses artefatos foram fornecidos templates e guidelines de como utilizá-los.

Para avaliar a utilidade do processo e do jogo os alunos responderam um questionário online. Os alunos apontaram que o processo auxiliou na GCS dos artefatos produzidos e que apesar dos templates fornecidos facilitarem sua execução, necessitavam de melhorias, pois alguns campos ficaram ambíguos, tais como a "necessidade da mudança" e "justificativa de mudança", presentes no "Formulário de Solicitação de Mudança". Os alunos também reportaram que o layout das tabelas fornecido no Formulário e no Relatório de Mudança dificultou a leitura do documento.

Além dos resultados do questionário, os artefatos que os alunos produziram referentes ao processo STONE foram avaliados e alguns problemas foram encontrados. Foram detectadas inconsistências na identificação dos itens de configuração, os alunos tiveram dificuldade de gerar um identificador único para cada versão do artefato. Também foram detectadas múltiplas solicitações de mudança agrupadas como se fossem uma só, o que deveria ser evitado. A recomendação é que cada alteração no artefato deveria ser discriminada em diferentes solicitações de mudança para facilitar a identificação e reparação dos problemas encontrados.

Apesar dos problemas reportados no questionário e na validação dos artefatos de GCS produzidos, observou-se que todos os alunos compreenderam o objetivo dos artefatos para o processo. Também foram apontados pelos alunos como pontos positivos do processo que a elaboração dos artefatos ajudou a manter e controlar as mudanças nos artefatos produzidos durante a disciplina.

À respeito do jogo, os alunos apontaram que a dinâmica foi desafiadora e animada, no entanto, foi um pouco confusa, já que as regras do jogo e os papéis dos jogadores não ficaram muito claros. Também foi mencionado que o jogo despertou o interesse do aprendizado ao processo STONE, pois a explicação do processo somente através da aula expositiva não é tão atrativa. Uma sugestão dos alunos foi a elaboração de um

manual para explicar mais detalhadamente a dinâmica do jogo. Outra sugestão foi melhorar sua jogabilidade, já que a execução das etapas não foi tão intuitiva.

A seguir são listadas algumas lições aprendidas em relação ao uso do processo STONE.

LA01: O jogo foi um facilitador para o entendimento do processo, além de estimular o interesse dos alunos nas atividades de GCS. Os alunos mostraram bastante interesse durante o jogo para entender as suas regras e os conceitos de GCS relacionados. Porém, observou-se durante a dinâmica a necessidade de melhorar a jogabilidade e o fornecimento de um manual para que os alunos pudessem entender melhor a dinamicidade do jogo.

LA02: O fornecimento de templates de GCS e exemplos facilitaram a execução das atividades do processo. Contudo, os templates precisam ser refinados, pois a avaliação mostrou que eles não foram claros o suficiente para os alunos e podem ter induzido-os à erros no preenchimento.

LA03: A definição de papéis e a sugestão de recursos/ ferramentas livres foram importantes para o uso do processo. A maioria dos alunos não têm conhecimento das ferramentas de GCS principalmente as que dão apoio às atividades de planejamento. Além disso, a definição de papéis entre os alunos foi essencial para distribuir as atividades de GCS. Por exemplo, um responsável para implementar a mudança e armazenar os IC alterado no repositório, este definido no Plano de GCS. Outro responsável por averiguar se a mudança foi implementada conforme o Formulário de Solicitação de Mudanças.

Em relação às ferramentas sugeridas, foi feita uma avaliação em uma disciplina optativa de "Tópicos Avançados em Engenharia de Software" com 4 alunos das ferramentas Mantis e GitLab. O objetivo era analisar o uso dessas ferramentas nas atividades da fase de "Controle de Mudanças". Inicialmente, foi conduzido um tutorial das ferramentas e foi fornecido um cenário no qual os alunos desempenharam atividades usando ambas as ferramentas. A avaliação do uso das ferramentas foi feita através de um questionário. Os resultados mostraram que os participantes concordaram totalmente (2 alunos) ou parcialmente (2 alunos) que ferramenta GitLab melhorava o desempenho bem como a produtividade das atividades realizadas. Em relação à facilidade e experiência de uso, a maioria dos participantes também apontaram o GitLab em vez do Mantis.

6.1 **Ameaças à Validade**

Dentre as atividades do processo e a sua aplicação uma limitação diz respeito à definição do processo para atender às necessidades dos alunos no âmbito acadêmico. Para minimizar essa ameaça foi feito um estudo nas normas técnicas e modelos de qualidade que tratavam GCS para que as atividades básicas fossem contempladas no processo, porém, adaptadas aos projetos acadêmicos. Além disso, foi feita uma pesquisa através de um questionário aplicado aos alunos do curso de CC e áreas afins para que as suas necessidades fossem realmente identificadas. Embora, a quantidade de participantes possa não ter sido expressiva (67 alunos), os alunos eram de cursos, de semestres e de campi diferentes da universidade. Outra limitação refere-se à utilização do processo em apenas uma disciplina. Para minimizar essa ameaça, foi

selecionada uma turma de graduação de uma disciplina optativa em que os alunos apresentam um maior interesse nas atividades extraclasse. Dessa forma, houve adesão da maioria da turma no uso do processo bem como na elaboração dos artefatos requeridos.

7 Trabalhos Relacionados

Na literatura, diversos trabalhos são propostos para a GCS, e dentre estes foram analisados os trabalhos que apresentavam soluções de GCS em contexto acadêmico [2][4][5][6][12][14][16]. Alguns desses trabalhos são detalhados a seguir.

Conn [4] propõe um processo de Engenharia de Software direcionado para projetos acadêmicos com o intuito de assegurar a qualidade de diferentes artefatos, tais como código, especificações, plano de GCS. Dentre as atividades do projeto, o autor sugere o "Controle de Configuração", na qual o gerente de suporte mantém todos produtos criados ou modificados sob controle de configuração. Porém, o passo a passo para atingir este resultado não é apresentado. Diferente de Conn, este trabalho define atividades de GCS de forma detalhada mostrando o passo a passo para a execução bem com os recursos envolvidos.

Haley et al. [6] propõem a documentação de mudanças em projetos de software acadêmicos através de *wikis* com o intuito de promover uma plataforma de aprendizagem cooperativa entre os participantes. Embora os autores tratem de gerenciamento de mudanças em projetos de software nenhum processo é apresentado.

Ríos et al. [14] definem uma metodologia baseada em plataformas de desenvolvimento colaborativo e sistemas de controle de versão, como o GitLab, para auxiliar os alunos a resolverem tarefas. Diferente dos trabalhos de [4] e [6], os autores dão ênfase em controle de versão e integração contínua aplicados a projetos desenvolvidos na academia, contudo os autores não abordam a GCS nas fases de planejamento e controle de mudanças como é proposto neste artigo.

Outros trabalhos de GCS no âmbito acadêmico [2][5][12][16] dão ênfase no uso de ferramentas (Git) e repositórios (GitHub) para controle de versão de código, não abrangendo as atividades relacionadas ao planejamento e controle de mudanças.

8 Conclusão e Trabalhos Futuros

Neste artigo foi proposto um processo de GCS para auxiliar os alunos em seus projetos acadêmicos. A metodologia consistiu em identificar as necessidades dos alunos em relação à GCS e suas atividades, bem como investigar como a GCS era aplicada pelos alunos em seus projetos acadêmicos. A primeira versão do processo foi definida com base na pesquisa da literatura e na pesquisa (*survey*) realizada com os 67 alunos.

A avaliação do processo foi conduzida em uma turma de graduação do curso de CC, e suas atividades e artefatos foram exemplificados através de um jogo com o intuito de facilitar o entendimento dos alunos à respeito de como utilizar o processo. Após a aplicação do jogo, os alunos utilizaram o processo para modificar as

alterações nos seus projetos de disciplina. Os resultados iniciais foram satisfatórios em relação ao uso do jogo, das atividades do processo, dos templates e dos recursos fornecidos. No entanto, os alunos sugeriram melhorias tanto para o processo quanto para o jogo. Como trabalhos futuros, pretende-se aplicar o processo em outras turmas de graduação e, com base nas lições aprendidas, pretende-se refinar o processo e melhorar a dinâmica do jogo para que este possa facilitar a sua aplicação. Além disso, pretende-se automatizar alguns passos do processo para que os alunos possam utilizá-lo de maneira mais eficiente.

Referências

1. Andrade, R. M., Marinho, F. G., Leitão, V. L., & Rocha, L. S. (2008). Uma Proposta de Metodologia para o Ensino de Engenharia de Software. Fórum de Educação em Engenharia de Software (FEES).
2. Arora, R., Goel, S., & Mittal, R. K. (2017, August). Supporting collaborative software development in academic learning environment: A collaborative pair and quadruple programming based approach. In 2017 Tenth International Conference on Contemporary Computing (IC3) (pp. 1-7). IEEE.
3. Comissão de Estudo de Gestão de Configuração. NBR ISO 10007: Gestão da qualidade - Diretrizes para a gestão de configuração. Rio de Janeiro, 1996.
4. Conn, R. (2004, March). A reusable, academic-strength, metrics-based software engineering process for capstone courses and projects. In ACM SIGCSE Bulletin (Vol. 36, No. 1, pp. 492-496). ACM.
5. Feliciano, J., Storey, M. A., & Zagalsky, A. (2016, May). Student experiences using GitHub in software engineering courses: a case study. In Proceedings of the 38th International Conference on Software Engineering Companion (pp. 422-431). ACM.
6. Haley, Elise R.; Collins, Galen B.; Coe, David J. The wonderful world of wiki benefits students and instructors. IEEE Potentials, v. 27, n. 2, 2008.
7. Humble, J., & Farley, D. (2010). Continuous Delivery: Reliable Software Releases through Build, Test, and Deployment Automation (Adobe Reader). Pearson Education.
8. IEEE Computer Society. 2001. Guide to Software Engineering Body of Knowledge (SWEBOK). Los Alamitos: CS Press.
9. IEEE Std. 1042. Guide to Software Configuration Management: An American National Standard. IEEE, 1988.
10. ISO/IEC/IEEE 24765:2009, Systems and Software Engineering–Vocabulary.
11. Krusche, S., Alperowitz, L., Bruegge, B., & Wagner, M. O. (2014). Rugby: an agile process model based on continuous delivery. RCoSE, 14, 42-50.
12. Mäkiaho, P., Poranen, T., & Seppi, A. (2014, June). Version control usage in students' software development projects. In Proceedings of the 15th International Conference on Computer Systems and Technologies (pp. 452-459). ACM.
13. Molinari, L. (2007). Gerência de Configuração-Técnicas e Práticas no Desenvolvimento do Software. Florianópolis: Visual Book.
14. Ríos, Julio César Cortés et al. A methodology for using GitLab for software engineering learning analytics. In: Proceedings of the 12th International Workshop on Cooperative and Human Aspects of Software Engineering. IEEE Press, 2019. p. 3-6.
15. SOFTEX. MR-MPS-SV – Guia de Implementação – Parte 2:2013.
16. Tirkey, A., & Gary, K. A.. Curricular change management with Git and Drupal: A tool to support flexible curricular development workflows. In 2017 IEEE 15th International Conference on Software Engineering Research, Management and Applications (SERA) (pp. 247-253). IEEE.