Uma Abordagem para o Processo de Gerenciamento de Configuração de Software

João Ronaldo Del-Ducca Cunha, Antonio Francisco do Prado, Antonio Carlos dos Santos, Raphael Marcílio de Souza Neto

Departamento de Computação, Universidade Federal de São Carlos, Rodovia Washington Luiz km 235, São Carlos, SP, Brasil
{jronaldo, prado, santos, raphael}@dc.ufscar.br

Resumo - O Gerenciamento de Configuração de Software (GCS) é um processo que provê recursos para identificação, controle da evolução e auditagem dos artefatos de software criados durante o desenvolvimento do projeto de software. No entanto, a sua aplicação em empresas de software é complexa, exigindo gastos que muitas vezes a inviabilizam. Uma forma de reduzir os custos envolvidos nesta aplicação está na definição de uma abordagem para o processo de GCS que englobe aspectos deste processo de tal forma a atender as necessidades específicas da empresa. Apresenta-se, neste artigo, uma abordagem para o processo de GCS, definindo atividades que buscam atender a área de processo de Gerenciamento de Configuração do modelo CMMI. Esta abordagem está sob o contexto de uma estratégia de implantação do processo de GCS e possui uma ferramenta de GCS para sua implementação.

Palavras-chave: Gerenciamento de Configuração de Software, Melhoria de Processos de Software.

Abstract - The Software Configuration Management (SCM) is a process that provides resources for identification, control of evolution and audit of the software's artifacts during the software's project development. However, it is complex to adopt the SCM process in software companies, because it demands investments that, in some cases, unable it. An option to reduce the costs involved in its deployment is the use of an approach of SCM process that includes in its activities aspects that answer the specific needs of a company. This article presents a SCM process approach where its phases answer the critical needs of the CMMI's (Capability Maturity Model Integration) Configuration Management process. This approach is part of a SCM implementation strategy and is bringing among its results an application tool to implement a SCM process.

Key-words: Software Configuration Management, Software Process Improvement.

Introdução

Muitos esforços foram realizados, por meio de estudos e pesquisas que visam contribuir para a evolução da Engenharia de Software e de suas áreas de conhecimento através da definição de processos de software. Dentre os processos de software, cita-se o de Gerenciamento de Configuração de Software (GCS) que está relacionado à Gestão da Qualidade [1], contribuindo para a qualidade do projeto de software através do suporte às suas atividades relacionadas tanto aos aspectos gerenciais quanto técnicos.

Além disso, outras pesquisas foram e estão sendo realizadas para definir modelos, normas e padrões para qualidade de processos de software como, por exemplo, o GCS, com o objetivo de direcionar os esforços realizados por organizações, como as de tecnologia da informação, para melhorar a qualidade no desenvolvimento de software e, por conseqüência, aumentar o grau de sucesso dos projetos de software. Cita-se, como exemplo, o CMMi (*Capability Maturity Model Integration*) [2] definido pela SEI (*Software Engineering Institute*). Com base nestas pesquisas, os processos existentes foram aprimorados e novos foram elaborados de tal forma a serem aplicados em empresas de software.

No entanto, ainda é notória a dificuldade das organizações em adotar conceitos e práticas relacionadas ao processo de GCS. Esta dificuldade muitas vezes reside no fato de que a implantação deste processo é uma atividade complexa que envolve definir uma abordagem adequada para o processo de GCS e a seleção de ferramentas de apoio a esta abordagem. Por este motivo, implantar o GCS exige investimentos tanto em recursos financeiros como humanos que são escassos principalmente para pequenas organizações, o que acarreta muitas vezes num impedimento à adoção dessas práticas que poderiam contribuir para o seu crescimento e sobrevivência.

Este artigo apresenta uma abordagem para o processo de GCS, definida sob o contexto de uma estratégia para implantação deste

processo, cujo objetivo é contribuir para com a melhoria da qualidade dos projetos de software e, conseqüentemente, com o sucesso das empresas de software frente o mercado de informática, facilitando a aplicação deste processo.

Processo de GCS

A finalidade do GCS é estabelecer e manter a integridade dos produtos do projeto de software ao longo do seu ciclo de vida. O GCS envolve [3]:

- Identificar a configuração de software (artefatos de software selecionados e sua descrição) em um dado momento;
- Controlar sistematicamente as mudanças na configuração;
- Manter, através da gerência, a integridade e rastreabilidade da configuração ao longo do ciclo de vida do software;
- Controlar a integridade de artefatos compostos, levando em conta a versão de cada um dos componentes, ou seja, controlar a configuração do artefato composto; e
- Registrar e relatar o estado do processo de alteração.

Os artefatos controlados que fazem parte da configuração do software são tidos como Itens de Configuração (*Configuration Itens – CIs*) e só podem ser alterados segundo os procedimentos definidos pelo processo de GCS. Estes Itens de Configuração formam linhas de base (*baselines*) que norteiam as próximas atividades a serem executadas no desenvolvimento do projeto.

Com base nestes conceitos, o GCS define atividades cuja realização elevam a confiança e a qualidade do software provendo [4]:

- Uma estrutura para identificação e controle dos CIs: documentação, código, interfaces, bancos de dados, entre outros; e
- Informações de gerenciamento do projeto e do produto preocupando-se com o controle das *baselines*, das mudanças, dos testes, das distribuições, das auditorias, entre outras atividades de desenvolvimento de um projeto de software.

Essas atividades de GCS são [5]: Identificação da Configuração (*Configuration Identification*); Controle da Configuração (*Configuration Control*); Administração de Estado (*Status Accounting*); e Auditagem da Configuração (*Configuration Audit*).

A Identificação da Configuração provê a infra-estrutura para as demais atividades do GCS e envolve, por exemplo, a aquisição de CIs e o estabelecimento de *baselines*. O Controle de Configuração define atividades a serem executadas para realizar modificações sobre os CIs do projeto. A Administração de Estado está relacionada à geração de relatórios com base nas informações da configuração do software.

Já a Auditagem da Configuração está relacionada às auditorias realizadas sobre CIs e *baselines*, buscando verificar se há conformidade entre características físicas e funcionais do software representadas pela documentação do projeto e pelo produto de software.

Implantação de GCS

O sucesso do projeto de software está muitas vezes associado com a adoção de processos de software como o de GCS. Porém, a adoção desses processos é uma tarefa complexa que muitas vezes acaba por inviabilizar as empresas em sair do diagnóstico, onde se observa a necessidade em implantar o processo, para chegar no operacional, onde se tem a execução destes processos nos projetos de software.

Isto se deve muitas vezes ao fato de se acreditar que a implantação de processos de software em uma empresa de software, como o caso da implantação do processo de GCS, ocorre simplesmente pela adoção de ferramentas. No entanto, a maior parte dos custos envolvidos nesta implantação relacionam-se a gastos com recursos humanos e na estruturação organizacional.

O processo de GCS apóia os demais processos de desenvolvimento de software e está relacionado a diversas atividades realizadas durante o desenvolvimento de um projeto de software. Portanto, implantá-lo implica em afetar vários setores da organização. Além disso, os processos de GCS, no período de implantação, podem acarretar uma maior lentidão do processo de desenvolvimento de software, incorporando práticas muitas vezes vistas como burocráticas. Assim, seu sucesso acaba dependendo mais de aspectos culturais do que dos técnicos, gerenciais e organizacionais. Os seguintes aspectos estão envolvidos na implantação de GCS [6]:

- **Aspectos Gerenciais:** Como planejar o processo, acompanhar as modificações, estabelecer prioridades e cronogramas;
- **Aspectos Organizacionais:** Infra-estrutura da empresa, autoridades e responsabilidades necessárias para a execução do processo;
- **Aspectos Culturais:** Como as pessoas lidam com a qualidade de software, qual a cultura existente na empresa e qual a maneira mais adequada de se fazer mudanças em tal cultura; e
- **Aspectos Técnicos:** Qual ferramenta adotar, como configurá-la, entre outros.

Devido a estes aspectos e aos custos, muitas empresas, a maioria de pequeno porte,

acabam por não conseguir implantar o GCS. Como o GCS é um processo que auxilia na qualidade do projeto, isso acaba prejudicando a economia relacionada ao desenvolvimento de software, pois muitas destas empresas não conseguem se firmar no mercado de informática.

Neste contexto, para facilitar a implantação do processo de GCS, a empresa deve planejar as atividades a serem realizadas durante a implantação. Dentre os estudos e pesquisas relacionados com implantação e melhoria de processos, cita-se o modelo IDEAL (acrônimo formado pelas iniciais das fases definidas no modelo) [7], definido pela SEI, instituto que estabeleceu o SW-CMM (*Capability Maturity Model for Software*) [3] e o CMMi (*Capability Maturity Model Integration*) [2]. O IDEAL é um modelo de programas de melhoria que pode ser utilizado como mecanismo para realizar a melhoria dos processos de software [1]. Este modelo propõe que a melhoria nos processos de software ocorre através de ciclos de melhoria onde, em cada ciclo, é executado um conjunto de atividades conforme mostra a Figura 1. Estas atividades são agrupadas nas cinco fases do modelo [1]:

- **Inicio (*Initiating*):** Lançar as bases para a realização bem sucedida do ciclo. Para tanto, têm-se as atividades como a da Motivação, onde a empresa é levada a querer realizar a melhoria, a de Patrocínio, onde se obtém o apoio de pessoas de dentro da empresa, e a de Infra-Estrutura, onde se tem, principalmente, a seleção das pessoas que executarão as demais atividades do ciclo de melhoria do modelo IDEAL;
- **Diagnóstico (*Diagnosting*):** Determinar onde se está e aonde se quer chegar. Para tanto, têm-se as atividades como a de Aferição, onde é caracterizado tanto o estágio atual como o estágio onde se quer chegar, e a de Recomendações para resolver possíveis problemas detectados durante a aferição;
- **Estabelecimento (*Establishing*)**: Planejar os detalhes de como chegar ao destino. Para tanto, têm-se as atividades como a de Priorização, onde se prioriza os esforços da melhoria considerando a realidade da empresa, a de Abordagem, onde são definidos os passos para realização da melhoria, e a de Planejamento que envolve a elaboração de um plano detalhado de realização considerando prazos, custos, riscos, pontos de controle responsabilidades e outros elementos;
- **Ação (*Acting*):** Executar o plano. Para tanto, têm-se as atividades como a de Criação da Solução, onde se reúnem os elementos da solução como padrões, modelos, ferramentas e treinamento, a de Testes-Piloto para testar a solução, a de Refinamento, onde é realizado o aperfeiçoamento da solução com base no que foi aprendido nos testes-piloto, e a de Implantação Completa, onde a solução é transferida para o restante dos projetos da empresa; e
- **Lição (*Leveraging*):** Aprender com a experiência. Para tanto, têm-se as atividades como a de Análise e Validação, onde são feitas coletas, análises, checagens e documentação das lições aprendidas, e a de Proposição de Ações Futuras, onde são desenvolvidas recomendações para o próximo ciclo de melhoria de processos. A tradução Lição para a palavra *Leveraging* foi utilizada para que a letra inicial de casa fase, cujo nome foi traduzido para o Português, forme também o mesmo acrônimo que, em Inglês, dá nome ao modelo.

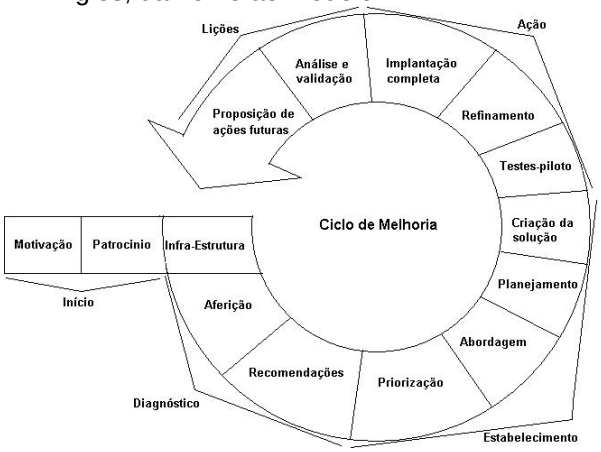

Figura 1 – Modelo IDEAL [8]

Com base no modelo IDEAL, em estudos referentes à implantação de GCS, como por exemplo, [9,10], e focando a realidade das pequenas empresas, definimos uma estratégia para implantação desse processo cujas atividades são:

1. Busca de Patrocínio.
2. Levantamento de Informações sobre a Empresa.
3. Levantamento de Informações sobre o processo de GCS.
4. Estabelecimento de Objetivos e Metas.
5. Definição de uma Abordagem para o Processo de GCS.
6. Seleção de Ferramentas de GCS.
7. Escolha de um Projeto-Piloto.
8. Definição do Plano de GCS.
9. Implantação do Plano de GCS.
10. Acompanhamento da Execução do Plano de GCS.
11. Avaliação dos Resultados Obtidos na Implantação do GCS.
12. Extensão do GCS para os demais Projetos da Empresa.

Dentre estas atividades, cita-se: a "Definição de uma Abordagem para o Processo de GCS" que atenda as necessidades da organização onde o processo está sendo implantado; e a "Seleção de Ferramentas de GCS" que dêem suporte a esta abordagem.

Conceitos da Abordagem de GCS Proposta

A abordagem baseia-se nos conceitos de GCS que o definem como sendo um processo gerencial de software, integrante do processo de Gerenciamento da Qualidade, cujo objetivo é estabelecer e manter a integridade dos resultados de um projeto de software, tanto intermediários como finais, ao longo do seu ciclo de vida. Este processo oferece suporte e auxilio as atividades executadas no decorrer do projeto contribuindo para o seu desenvolvimento e posterior manutenção.

Segundo os conceitos definidos na abordagem, um projeto (*Project*) é uma unidade de gestão na qual se realiza algo único, que possui uma data de início e término [11], e que representa a execução de processos de software. Este projeto pode ser dividido em módulos (*Modules*), para facilitar o seu gerenciamento, e possui uma configuração de software que representa o conjunto de características físicas e funcionais do software conforme estabelecido em algum documento técnico ou realizado por um produto desenvolvido no projeto [12].

Esta configuração é composta por artefatos (*Artifacts*) e CIs. Ambos são informações criadas durante o desenvolvimento de um produto de software, ou que sejam necessários para seu desenvolvimento [13]. Além disso, um CI é um artefato:

- Concluído;
- Entregue e avaliado segundo atividades definidas pelo GCS para compor uma determinada versão da configuração do projeto; e
- Controlado, pois só poderá ser alterado após uma aprovação prévia definida por atividades do processo de GCS.

Além disso, a abordagem estende o conceito de CI e artefatos. Assim, um artefato pode ser definido como sendo: um Produto (*Product*) que corresponde a um arquivo do projeto; um Item Interno (*Internal Item*) que corresponde a itens internos do artefato; ou um Relacionamento (*Relationship*) que corresponde a um relacionamento entre artefatos produto, artefatos item interno ou entre artefatos produto e item interno. Este conceito permite um controle dos artefatos em diferentes níveis de abstração.

Neste contexto, a cada categoria de artefato (Produto, Item Interno ou Relacionamento) são associados tipos que o descrevem como, por exemplo, classes, documento de especificação de requisitos, entre outros. Além da descrição, cada tipo também possui um critério de aprovação a ser adotado nas atividades da abordagem proposta.

Tanto os artefatos quanto os CIs são armazenados numa estrutura própria para o projeto chamada de biblioteca de software (*Software Library*) do projeto. Assim, define-se biblioteca de software como sendo uma coleção controlada de software e documentos relacionados que auxilia o desenvolvimento, uso e manutenção do software [12], sendo também um instrumento utilizado para realizar atividades de distribuição e entrega do software [5]. As técnicas e métodos usados pelo GCS geralmente estão centrados no controle dessas bibliotecas [14] que podem ser categorizadas de acordo com o grau de maturidade e significado dos seus elementos [5]. Dentre os possíveis tipos de biblioteca, existem: as de desenvolvimento (*Development*) que estão diretamente relacionados com os módulos do projeto e onde são armazenados os artefatos que fazem parte do desenvolvimento do projeto e que ainda não são controlados; e as de produção (*Master*) onde são armazenados os CIs.

Os CIs compõem, em pontos de controle definidos durante o ciclo de vida do projeto, *baselines* que determinam uma versão do software desenvolvido no projeto [12] naquele momento. Esta *baseline* tem como objetivo servir de referência para as próximas atividades do desenvolvimento do projeto sendo, no entanto, apenas um conceito lógico composto pelos CIs do projeto. Toda *baseline* está associada a um critério de aprovação (*Baseline Acquisition Criterion*) a ser adotado na atividade de "Aquisição da Baselines", da abordagem.

Agentes de Negócio da Abordagem de GCS Proposta

O processo de GCS, por estar relacionado a diversos ambientes de desenvolvimento dentro de um projeto de software, apresenta agentes de negócio já existentes dentro de uma empresa. Segue os agentes de negócio definidos para esta abordagem:

- **Gerente do Projeto (*Project Manager*):** É aquele que definiu o projeto e que coordena as atividades relacionadas com as áreas de conhecimento de gestão de projetos como, por exemplo, a definição da equipe, o planejamento e controle de tarefas e atividades do projeto;
- **Engenheiro de Software (*Software Engineer*):** É quem realiza as implementações e manutenções dos artefatos do projeto. Pode vir a desempenhar diferentes papéis de acordo com o que é estabelecido na organização, como por

exemplo, analista de sistemas, programador, integrador, entre outros;

- **Comitê de Controle de Configuração (*Configuration Control Board - CCB*):** É um comitê que dependendo da complexidade do projeto pode ser formado por uma ou várias pessoas. Normalmente é formado pelo Gerente do Projeto e os lideres de cada um dos módulos desenvolvidos no projeto;
- **Bibliotecário (*Librarian*):** É quem tem sob custódia e controle a biblioteca de software do projeto e os artefatos nela armazenados;
- **Responsável pelo Gerenciamento de Configuração (*Configuration Management Officer - CMO*):** É quem executa as atividades operacionais do GCS como, por exemplo, o controle dos formulários criados durante as atividades. Dependendo do tamanho do projeto, é imprescindível o apoio de ferramentas para auxiliar o CMO em suas responsabilidades. Inclusive as atividades, sob responsabilidade deste papel, podem estar sendo automatizadas parcialmente ou completamente através do uso de uma ferramenta de GCS;
- **Comitê de Engenharia de Processo (*Process Engineer Board – PEB*):** É um comitê que define e controla o processo de desenvolvimento, evoluindo-o de acordo com as necessidades organizacionais;
- **Auditor (*Auditor*):** É quem está relacionado à qualidade de software e que possui responsabilidades definidas para as áreas de Qualidade adotadas na organização. Portanto, segue critérios de qualidade definidos para avaliar o que é realizado dentro do projeto; e
- **Comitê de Garantia de Qualidade (*Quality Assurance Board - QAB*):** É um comitê que define os critérios de aprovação associados aos CIs e *baselines*.

Cada um destes agentes de negócio possui responsabilidades definidas com base nas atividades de GCS apresentadas a seguir.

Dependendo da quantidade de recursos presentes na organização, pode ocorrer que membros da equipe desempenhem o papel de mais de um agente.

Atividades da Abordagem de GCS Proposta

A abordagem de GCS proposta define um conjunto de atividades relacionadas com a Identificação, Controle, Auditagem e Administração de Estado da Configuração do projeto de software.

Identificação da Configuração: Conforme mostra a Figura 2 as atividades de Identificação de Configuração compreendem:

- **Identificação de *Baselines* (*Baseline Identification*):** O Gerente do Projeto define, com base em informações gerenciais do projeto, quais *baselines* serão estabelecidas durante o desenvolvimento do projeto. A partir destas *baselines*, o QAB define os critérios de aprovação (*Baseline Acquisition Criterion*) que deverão ser aplicados para cada *baseline*.
- **Identificação de Tipos de CIs (*CI Types Identification*):** O PEB define os tipos de CI desenvolvidos na organização e que estarão disponíveis para a realização da atividade de Identificação de CIs. Conforme o conceito de CI adotado nesta abordagem, nesta atividade devem ser definidos os tipos de CIs Produto, Item Interno e Relacionamento que serão controlados. Para cada tipo definido pelo PEB, o QAB deve definir critérios para aprovação (*CI Acquisition Criterion*) das informações que correspondem a estes tipos.
- **Identificação de CIs (*CI Identification*):** O CCB define quais tipos de informações do projeto serão controladas e irão compor as *baselines* do projeto com base na lista de tipos de CIs definidos na atividade de Identificação de Tipos de CIs.
- **Identificação da Estrutura (*Structure Identification*):** o Gerente do Projeto define os módulos do projeto, e o Bibliotecário define a estrutura da biblioteca de desenvolvimento associada a cada módulo definido.
- **Aquisição de CIs (*CI Acquisition*):** Os Engenheiros de Software realizam a entrega das informações dos projetos para que estas se constituam em CIs. Estas informações são avaliadas pelo auditor com base nos critérios de aprovação de CIs (*CI Acquisition Criterion*) definidos na atividade de Identificação de Tipos de CIs.
- **Aquisição de *Baselines* (*Baseline Acquisition*):** O Gerente do Projeto solicita a aquisição da *baseline* que é avaliada por um Auditor com base nos critérios de aprovação (*Baseline Acquisition Criterion*) definidos na atividade de Identificação de *Baselines*. Além disso, o Gerente do Projeto pode abrir uma nova *baseline* ou concluir o projeto.

Controle da Configuração: Conforme mostra a Figura 3 as atividades de Controle de Configuração compreendem:

- **Requisição de Mudança (*Request Change*):** O Engenheiro de Software relata um problema detectado ou uma melhoria requerida em uma dada *baseline* do projeto de software, preenchendo um formulário para requisição de mudanças (*Change Request Form – CRF*).

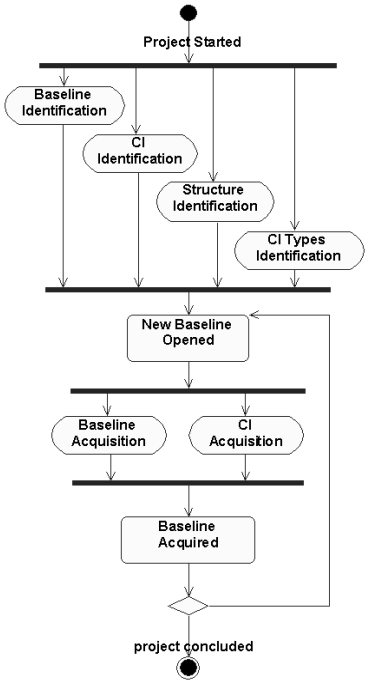

Figura 2 – Atividades de Identificação da Configuração

- **Solicitação de Modificação (*Solicit Change*):** O CCB avalia as requisições de mudança quanto a sua pertinência, seleciona aquelas similares para que possam compor uma solicitação de modificação e preenche um formulário para realizar esta solicitação (*Change Solicitation Form – CSF*). Além disso, o CCB define uma equipe de manutenção, formada por Engenheiros de Software, que irá ser responsável por planejar e realizar a modificação.
- **Planejamento da Modificação (*Plan Modification*):** A equipe de manutenção planeja a modificação, preenchendo um Plano de Modificação (*Modification Plan – MP*) com base na solicitação de modificação e, neste planejamento, selecionam os CIs da *baseline* a serem alterados. Para tanto, o CMO realiza a análise de impacto das mudanças e o Bibliotecário disponibiliza os CIs da biblioteca *Master* do projeto.
- **Implementação de Mudanças (*Implement Changes*):** a equipe de manutenção realiza as mudanças necessárias sobre os CIs, denominados artefatos em manutenção, presentes no Plano de Modificação. No momento em que a implementação é concluída, o Auditor avalia os artefatos em manutenção segundo os critérios de aprovação (*CI Acquisition Criterion*). Uma vez aprovados, estes CIs passam novamente a compor a *baseline* do projeto.

Auditagem da Configuração: As auditagens da configuração são muitas vezes dependentes do contrato que rege o projeto e devem ser bem planejadas e definidas podendo requisitar a

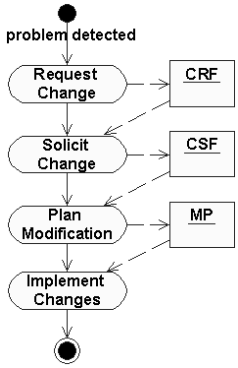

Figura 3 – Atividades de Controle da Configuração

atuação de vários profissionais que irão avaliar a configuração do projeto. Elas podem ser realizadas, portanto, em diferentes momentos do desenvolvimento e envolvem o trabalho do auditor, profissional não envolvido com o desenvolvimento do projeto mas que avalia a qualidade e conformidade do que está sendo construído.

Administração de Estado da Configuração: Por fim, a atividade de administração de estado gera uma série de relatórios sobre as demais atividades da abordagem e sobre a configuração do projeto. Dentre os relatórios, destacam-se:

- **Relatório sobre Tipos (*Types Report - TR*):** Lista todos os tipos de CI da empresa. Além disso, apresenta uma lista de tipos de CIs de cada um dos projetos.
- **Relatório sobre Artefatos (*Artifacts Report - AR*):** Lista de todos os artefatos, com suas respectivas versões (indicando quais deles são CIs) e todas as aquisições de CIs realizadas sobre eles.
- **Relatório de Mudanças (*Changes Report - CR*):** Lista todas as requisições de mudança, além das solicitações e planejamentos de modificação gerados para um dado projeto.
- **Relatório de *Baselines* (*Baselines Report - BR*):** Lista todas as *baselines* de um dado projeto, assim como os CIs que os constituem.
- **Relatório de CIs (*CIs Report - CIR*):** Apresenta informações sobre: a evolução de todos os CIs, mostrando o seu histórico; as atividades de Aquisição de CIs realizadas; os planos de modificação relacionados a eles; e outros projetos que os utilizam.
- **Relatório de Auditagens de Configuração (*Configuration Audit Report - CAR*):** Apresenta informações das auditagens realizadas na configuração do projeto, os CIs e *baselines* envolvidos, além do resultado associado a cada um deles (aprovado ou reprovado).
- **Relatório de Projetos (*Projects Report - PR*):** apresenta informações relacionadas à

configuração do projeto específico, com todas as suas *baselines*, CIs e distribuições.

Aderência ao modelo CMMI

O CMMI é um modelo de maturidade e capacitação que tem como objetivo a melhoria contínua dos processos de software. Para tanto, este modelo define áreas de processo (*Process Areas - PAs*) utilizadas para aferir a qualidade dos processos organizacionais, e definir a maturidade das organizações de software em desenvolver projetos de software.

A maturidade organizacional é dividida, segundo este modelo, em cinco níveis seguindo uma arquitetura em estágios [2]: Inicial (*Initial*), Gerenciado (*Managed*), Definido (*Defined*), Gerenciado Quantitativamente (*Quantitatively Managed*) e Otimizado (*Optimizing*).

No caso do nível gerenciado, ele indica que a organização já possui alguma maturidade no desenvolvimento de projetos de software. Existem políticas e regras definidas para o projeto, além de abordagens de implantação para cada um dos processos utilizados. Além disso, o planejamento e gerenciamento de projetos nestas organizações se baseiam no sucesso de outros projetos já desenvolvidos. Portanto, os projetos de organizações deste nível são gerenciados e os processos são planejados, realizados e controlados.

Para que uma organização atinja maiores níveis de maturidade, o CMMI estabelece que esta deve atender PAs as quais definem metas a serem cumpridas.

Com exceção do nível inicial, todos os demais níveis apresentam vários PAs. A Tabela 1 mostra as do nível gerenciado do modelo CMMI.

Process Area
Requirements Management
Project Planning
Project Monitoring and Control
Supplier Agreement Management
Measurement and Analysis
Project and Product Quality Assurance
Configuration Management

Tabela 1 – PÁS do nível Gerenciado

Para que uma empresa possa ser considerada madura segundo um dos níveis apresentados no CMMI, esta deve cumprir todas as metas definidas pelas PA daquele nível e dos anteriores. Estas metas prevêem uma série de práticas para nortear os esforços para a sua aplicação no processo de desenvolvimento de software da organização.

Uma das PAs do nível gerenciado do CMMI é a de Gerenciamento de Configuração que define práticas a serem atendidas pelas abordagens para os processos de GCS. Esta PA define três metas específicas e cada uma dessas metas prevê práticas que descrevem a infra-estrutura necessária e as atividades que deverão ser desempenhadas pela organização para atender a PA de GCS. A Tabela 2 mostra as atividades da abordagem proposta relacionadas às práticas de cada uma das metas da PA de GCS do CMMI.

Metas	Práticas da PA	Atividades da Abordagem
Estabelecer Baselines	Identificar CIs	Identificação de CIs; e Aquisição de CIs.
	Estabelecer um Sistema de Gerenciamento de Configuração	Definição da Estrutura do Projeto; e Definição de nomes
	Criar ou distribuir Baselines	Identificação de Baselines; e Estabelecimento de Baselines.
Rastrear e Controlar Mudanças	Rastrear as Requisições de Mudança	Requisição de Mudança; e Planejamento de Modificação.
	Controlar os CIs	Implementação da Mudança.
Estabelecer Integridade	Estabelecer Registros do Gerenciamento de Configuração	Aquisição de CIs; Identificação de Baselines; e Administração de Estado.
	Realizar Auditorias sobre a Configuração	Auditagem da Configuração.

Tabela 2 – Atividades da Abordagem Proposta, relacionadas com as Práticas da PA de GCS do CMMI

Implementação da Abordagem Proposta

Está sendo desenvolvida uma ferramenta para apoiar esta abordagem proposta, auxiliando as empresas que forem utilizá-la. Esta ferramenta, denominada SoCManager (*Software Configuration Manager*) oferece funcionalidades que permitem o GCS em granularidade fina (*fine-grained*) [15], segundo o conceito de CI apresentado na abordagem, dando apoio automatizado ao controle da evolução dos artefatos descritos segundo um meta-modelo baseado no MOF (*Metadata Object Facility*) [16] e representados segundo o padrão XMI (*XML Metadata Interchange*) [17].

A SoCManager está integrada ao ambiente *Orion* [18] que possui outras duas ferramentas de suporte aos processos técnicos de análise, projeto, implementação e testes de software: a MVCASE [19], para modelagem de sistemas; e a C-CORE [20] para implementação e testes. Esta integração ocorre através do uso de um repositório, controlado pela SoCManager, onde são armazenados os artefatos gerados pela MVCASE e C-CORE. Além disso, estes artefatos são representados pelo padrão XMI e, neste caso, a SoCManager realiza a introspecção sobre esses artefatos, através do uso do JMI (*Java Metadata Interface*) [21] e das bibliotecas do MDRRepository [22], facilitando a atividade de Aquisição de CIs sobre eles.

Os requisitos da SoCManager são oriundos de três fontes:

- Responsabilidades definidas para o agente CMO da abordagem para o processo de GCS apresentado neste artigo;
- Iterações dos demais agentes com as atividades de GCS da abordagem; e
- Aspectos que idealmente deveriam ser atendidos pelas ferramentas de GCS [5]: Bibliotecas de GCS; Requisição de Mudança de Software e Procedimentos de Aprovação; Tarefas de Gerenciamento de Mudanças; Reportar Status de Configuração de Software e Métricas de GCS; Auditoria de Software; Gerenciamento e Rastreamento das Documentações de Software; e Gerenciar e Rastrear as Liberações de Software e suas Distribuições.

Atualmente a SoCManager possui uma versão operacional e atende a atividade de Identificação de Configuração e parte das atividades de Controle de Configuração. O fato de a SoCManager estar fundamentada em uma abordagem para o processo de GCS torna os seus benefícios mais concisos e tangíveis.

A Figura 4 mostra a tela de definição da equipe do projeto onde o gerente define o papel de cada membro da equipe. Estes papéis correspondem aos papéis definidos na abordagem.

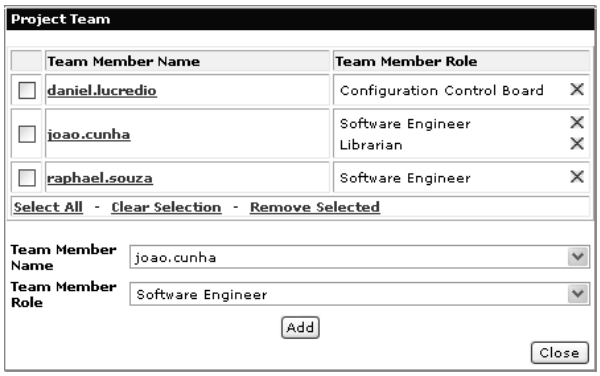

Figura 4 - Definição da Equipe do Projeto

Para cada papel desempenhado por um membro na SoCManager, há uma perspectiva associada a qual fornece, através do *Project Browser*, acesso aos serviços sob responsabilidade daquele papel. A Figura 5 mostra o *Project Browser* do bibliotecário que permite a definição dos repositórios onde serão armazenados os artefatos do projeto (correspondentes à biblioteca de desenvolvimento descrita na abordagem) e, para cada módulo definido no projeto, o local no repositório, onde serão armazenados seus artefatos.

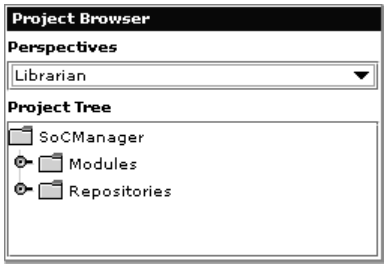

Figura 5 - *Project Browser* do bibliotecário

Trabalhos Relacionados

As abordagens para o processo de GCS normalmente atendem apenas parte das atividades deste processo e não tratam diretamente as necessidades das empresas de pequeno porte. Dentre as abordagens existentes, citam-se:

- **Abordagem ITI de GCS [6]:** desenvolvida pela ITI (Instituto nacional de Tecnologia da Informação) e aplicada numa pequena empresa. Esta abordagem define atividades relacionadas apenas com o Controle de Configuração;
- **Abordagem INPE de GCS [23]:** desenvolvida para ser aplicada em um projeto de desenvolvimento de software para controle de satélites do Instituto Nacional de Pesquisas Especiais (INPE). Define aspectos relacionados tanto a Identificação como o

Controle de Configuração, apresentando conceitos como o de Elemento Controlado adotado na abordagem para o processo de GCS proposta neste trabalho.

- **Abordagem Praxis de GCS [1]:** integrante do processo gerencial de Gestão da Qualidade e que está diretamente relacionado com a área chave de processo de Gerenciamento de Configuração do SW-CMM [3]. Não define explicitamente como realizar as atividades de Identificação, Controle, Auditagem e Administração de Estado da configuração do projeto de software.
- **Abordagem RUP de GCS [24]:** define uma série de atividades a serem desempenhadas por agentes para que se tenha a operacionalização deste processo. Porém, é difícil de ser adotada em empresas com recursos escassos, como as de pequeno porte, e não define explicitamente as atividades Identificação, Controle, Auditagem e Administração de Estado da configuração do projeto de software.
- **Abordagem GCONF de GCS [25]:** define um processo de GCS para ambientes de desenvolvimento de software orientados a organização (ADSOrg) que enfatizam a necessidade da gestão do conhecimento relacionado com a produção de software de uma determinada organização. Esse processo define gerentes de projeto e desenvolvedores como agentes que colaboram na execução de três atividades principais, que são: planejar o gerenciamento de Configuração, controlar liberação e distribuição e controlar configuração.

Além das abordagens apresentadas, existem outras que também tratam o processo de GCS. Dentre elas têm-se a da NASA [26], usada pelos centros de desenvolvimento de software dessa organização e pelos seus fornecedores de software.

Muitas dessas abordagens são executadas pelas empresas sem a preocupação com a aplicação de uma estratégia para implantação do processo de GCS. Além disso, não existem ferramentas de GCS que dêem apoio a todas as atividades definidas pelas abordagens. Estes fatores acabam dificultando a adoção do processo de GCS em empresas que desenvolvem software.

Já a abordagem apresentada neste artigo está sob o contexto de uma estratégia para implantação do processo de GCS e foi elaborada de tal forma que pudesse ser aplicado em empresas de pequeno porte. Além disso, a ferramenta SoCManager está sendo desenvolvida para automatizar as atividades dessa abordagem, facilitando com isso a adoção do processo de GCS.

Resultados

Visando obter resultados conclusivos quanto à aplicabilidade da Abordagem de GCS proposta, esta foi implantada, a partir da execução da estratégia de GCS descrita neste artigo, em uma empresa de software. Para tanto, foi escolhido um projeto de software em estágio inicial de desenvolvimento. A equipe do projeto era formada por 11 integrantes que foram treinados para desempenhar as atividades da abordagem de GCS.

Num primeiro momento, as dificuldades, inclusive relacionados ao conceito de GCS, foram evidentes. No entanto, a medida em que o processo foi sendo executado, foi possível torná-lo mais receptível, tanto aos membros do projeto como aos demais profissionais da empresa.

Para apoiar a aplicação da abordagem, foi utilizada inicialmente a ferramenta de controle de versões CVS (*Concurrent Version System*), além de e-mails para troca de informações relacionadas com as atividades de GCS da abordagem proposta.

No entanto, a partir do momento em que a ferramenta SoCManager foi disponibilizada, pode-se notar uma sensível melhora na execução da abordagem de GCS proposta.

Discussão e Conclusões

A proposta inicial desta pesquisa era o desenvolvimento de uma ferramenta de apoio ao processo de GCS. No entanto, para atingir este objetivo, foi elaborada uma estratégia de implantação do processo de GCS que define atividades que devem ser desempenhadas pelas empresas de software para tornar operacional o processo de GCS em seus projetos de software. A partir desta estratégia, foi definida uma abordagem que descrevesse o processo a ser automatizado pela ferramenta.

Este artigo apresentou esta abordagem, seus conceitos, agentes e atividades. A aplicação da Abordagem Proposta, descrita em Resultados, acabou por validar os objetivos iniciais da pesquisa, facilitando a aplicação do processo de GCS em empresas de software.

A melhoria da abordagem e, conseqüentemente, da ferramenta SoCManager, é algo contínuo e que depende da sua aplicação em projetos de software desenvolvidos por empresas de software. Por este motivo, a próxima etapa é analisar a aplicação da abordagem em outras empresas de software inclusive de maior porte.

Agradecimentos

A CAPES, pelo apoio financeiro.

Referências

1. Pádua F., W. P. Engenharia de Software: Fundamentos, Métodos e Padrões. LTC, 2003.
2. SEI. Capability Maturity Model Integration, v1.1. 2001. Disponível Site SEI. URL: (http://www.sei.cmu.edu/cmmi/). Consultado em 11/2004.
3. Paulk, M. C., Curtiss, W. B. e Chrissis, M. B. The Capability Maturity Model: Guidelines for Improving the Software Process. Addison-Wesley, 1999.
4. IEEE Std 828-1998: IEEE Standard for Software Configuration Management Plans. Standards Coordinating Committee – IEEE Computer Society, 1998.
5. IEEE. Guide to the Software Engineering Body of Knowledge 2004 Version. IEEE - Computer Society, Los Alamitos, USA, 2004.
6. Oliveira, A. A. C. P.; Formoso, P., F.;Cruz, J. L; Martino, W. R. Gerência de Configuração de Software: Evolução de Software sob Controle. ITI – Instituto Nacional de Tecnologia da Informação, Brasil, 2001.
7. Mcfeeley, R. IDEAL: A user guide for software process improvement. Relatório Técnico CMU/SEI-96-HB-001, Software Engineering Institute, 1996.
8. Gremba J.; Myers, C. The IDEAL model: A practical guide to improvement. Software Engineering Institute, 1997
9. Cugola, G.; Lavazza, L.; Nart, V. Manca; S.; Pagone, M. R.; Oliveira, A. L. C. An Experience in Setting-Up a Configuration Management Environment Software Technology and Engineering Practice, 1997. Proceedings., Eighth IEEE International Workshop on [incorporating Computer Aided Software Engineering] , 14-18 July 1997.
10. Visconti, M.; Guzmán, L. A Measurement-Based Approach for Implanting SQA & SCM Practices. Computer Science Society, 2000. SCCC '00. Proceedings. XX International Conference of the Chilean, 16-18 Nov. 2000.
11. Project Management Institute. PMBOK 2000:Project Management Book. Brazil Minas Gerais Chapter, 2002.
12. IEEE Std. 610.12-1990: IEEE Standard Glossary of Software Engineering Terminology. Standards Coordinating Committee - Computer Society, 1990.
13. Pressman, R., S. Software Engineering: A Practitioner's Approach. McGraw-Hill. 5a Edição. 2001.
14. ANSI/IEEE Std. 1042.-1987: IEEE Guide to Software Configuration Management. Technical Committee On Software Engineering - Computer Society, 1987.
15. Dittrich, K.R.; Tombros, D. and Geppert, A. Databases in Software Engineering: A Roadmap. In The Future of Software Engineering. ACM, New York, 2000, 291-302
16. OBJECT MANAGEMENT GROUP. Object Management Group. Meta-Object Facility (MOF) - Version 1.4. 2002. Disponível Site OMG. URL: (http://www.omg.org/technology/documents/formal/mof.htm) - Consultado em 2/2004.
17. OBJECT MANAGEMENT GROUP. XML Metadata Interchange (XMI) – Version 1.2. 2002. Disponível site OMG. URL: (http://www.omg.org/technology/documents/formal/xmi.htm). Consultado em 02/2004.
18. Lucredio, D.; Almeida E. S.; Bianchini, C. P.; Prado, A. F.; Trevelin, L. C. Orion – A Component Based Software Engineering Environment., in Journal of Object Technology. April, 2004.
19. Almeida, E. S.; Lucrédio, D.; Bianchini, C. P.; Prado, A. F.; Trevelin, L. C.; Ferramenta MVCASE - Uma Ferramenta Integradora de Tecnologias para o Desenvolvimento de Componentes Distribuídos. XVI Simpósio Brasileiro de Engenharia de Software, Sessão de Ferramentas, Brasil, 2002.
20. Souza N., R. M.; Lucredio, D.; Bossonaro, A.; Cunha, J. R. D. D.; Prado, A. F.; Catarino, I. C. S.; Souza, A. M. Component-Based Software Development Environment. 6th International Conference on Enterprise Information Systems - ICEIS - Porto – Portugal, 2004.
21. Sun. Java Metadata Interface (JMI). Disponível Site Sun. URL: (http://java.sun.com/products/jmi/index.jsp). Consultado em 7/2004.
22. NetBeans. Metadata Repository (MDR) Project Home. Disponível Site NetBeans. URL: (http://mdr.netbeans.org/). Consultado em 7/2004.
23. Cunha, J. B. S. Uma Abordagem de Qualidade e Produtividade para o Desenvolvimento de Sistemas de Software Complexos utilizando a Arquitetura de Placa de Software – SOFTBOARD. Tese de Doutorado em Computação Aplicada. Instituto Nacional de Pesquisas Espaciais – INPE. Abril, 1997.
24. Kruchten, P. Rational Unified Process: An Introduction. Addison-Wesley, Reading-MA, 2000.
25. Figueiredo, S. M. Gerência de Configuração em Ambientes de Desenvolvimento de Software Orientados à Organização, B.Sc., DCC - IM / UFRJ, Rio de Janeiro, RJ, 2004.
26. NASA Software Configuration Management Guidebook. 1997.Disponível Site NASA. URL: (http://satc.gsfc.nasa.gov/GuideBooks/cmpub.html) - Consultado em 1/2004.