Universidade do Estado de Santa Catarina
Centro de Educação Superior do Alto Vale do Itajaí

# DESENVOLVIMENTO COLABORATIVO NO ENTERPRISE ARCHITECT 11.1

Tiago Capistrano[1]
[1]Universidade do Estado Santa Catarina – UDESC Curso de Pós-Graduação
capistrano.tiago@gmail.com

**Resumo**
A colaboração é um processo consolidado e comprovadamente benéfico no desenvolvimento de códigos fonte de *softwares*, porém muito ausente no desenvolvimento de modelos de *softwares*. Com o aumento da complexidade dos *softwares* e as tendências apontando para o desenvolvimento orientado à modelos, o que podemos fazer para melhorar o processo de modelagem de *softwares*? A presente pesquisa busca na colaboração uma resposta para esta pergunta e com isso almeja facilitar o processo de iniciação da colaboração no desenvolvimento de modelos de *softwares* apresentando métodos de colaboração e de versionamento de modelos na ferramenta CASE *Enterprise Architect* versão 11.1, edição corporativa. A presente pesquisa evidenciou o potencial do desenvolvimento colaborativo de modelos e ao mesmo tempo sua imaturidade ocasionada pela complexidade destes e escassez de ferramentas, metodologias de trabalho e relatos de experiência que auxiliem esta metodologia de trabalho.

**Palavras-chave:** Modelagem Colaborativa.*Enterprise Architect*.

***Abstract***
*Collaboration is a consolidated and demonstrably beneficial in the development process of software source codes, but very absent in the development of software models. With the increasing complexity of software and trends pointing to the targeted development of models, what can we do to improve the process of software development models? This research seeks collaboration in an answer to this question and it aims to facilitate the process of initiation of the collaboration in the development of software models presenting methods of collaboration and versioning of models in CASE tool Enterprise Architect version 11.1, corporative edition. This research showed the potential of the collaborative development of models while their immaturity caused by the complexity of the models and the lack of tools, methodologies and work experience reports that help this work methodology.*

***Keywords:*** *Collaborative Modeling. Enterprise Architect.*

## 1. Introdução

A evolução tecnológica anseia por *softwares* em contextos cada vez mais abrangentes, ou seja, em diferentes dispositivos e plataformas, tornando gradativa a complexidade de desenvolvimento dos mesmos. Segundo Fuks (2012), uma abordagem para facilitar o desenvolvimento de *software* é a construção colaborativa dos modelos da Engenharia de *Software*. Ainda segundo Fuks (2012), modelos, ou artefatos, servem como abstrações do *software* que está sendo construído, como por exemplo, especificações de requisitos, diagramas de classes e diagramas de casos de uso.

Mais do que a boa vontade da gerência e dos profissionais da equipe, o trabalho colaborativo em organizações de desenvolvimento de *software* requer a formulação de processos e utilização de ferramentas que apoiem esta metodologia de trabalho. A grande maioria das organizações já trabalha de forma colaborativa no desenvolvimento do código fonte do *software*, utilizando, por exemplo, de *Integrated Development Environment* (IDE) integrado à ferramentas de controle de versão de documento. Já está mais do que consolidado o benefício dessas ferramentas no


ISSN 2316-4190, DOI: 10.5965/2316419004052015034

Universidade do Estado de Santa Catarina
Centro de Educação Superior do Alto Vale do Itajaí

processo de desenvolvimento do código fonte do *software*, mas quando se trata de ferramentas de apoio ao desenvolvimento colaborativo de modelos, ou as organizações desconhecem essas ferramentas ou não há interesse das mesmas pela questão da complexidade envolvida no processo.

Não é preciso analisar a fundo para notar que em grande parte das organizações não há trabalho colaborativo no processo de desenvolvimento da modelagem de *software*, muitas vezes esta etapa é realizada de forma individual, ou quando realizada em grupo é acompanhada de um processo burocrático. Muitas organizações já conhecem e utilizam ferramentas que dispõem de funcionalidades que auxiliam o trabalho colaborativo, principalmente no desenvolvimento do código fonte, mas muitas vezes o processo de modelagem do *software* não usufrui destas funcionalidades, ou essas organizações não possuem um processo definido de trabalho que permita o trabalho colaborativo na modelagem do *software*.

Segundo Teixeira (2009), a exigência crescente por projetos cada vez mais complexos atribui grande carga de responsabilidade à engenheiros de *software*, pois eles tem que ser capazes de executar tarefas colaborativas suportadas por computadores geograficamente distribuídos de forma mais eficaz possível. A prática do desenvolvimento distribuído de *softwares* é crescente, está cada vez mais comum encontrarmos equipes geograficamente distribuídas trabalhando no mesmo projeto de *software*. Segundo Prikladnicki (2010) apud Martinazzo (2011), dos diversos fatores que tem incentivado esta prática, pode-se destacar o fato de profissionais habilidosos poderem estar em diferentes locais ou até mesmo por questões de custo.

Outro fator que justifica a importância do presente artigo está na declaração de Pressman (2011), que afirma que o desafio para a próxima década é desenvolver métodos e ferramentas que facilitem essa colaboração. Esta afirmação de Pressman (2011) enaltece a carência do mercado de *software* em dispor de ferramentas que apoiam o trabalho colaborativo nas organizações, principalmente no que diz respeito a documentação e modelagem de artefatos de *software*.

Quanto a escolha da ferramenta de modelagem de *software Enterprise Architect*, esta se justifica por ser uma ferramenta flexível, que possui vários métodos de suporte ao trabalho colaborativo, o que propicia às organizações uma melhor aderência da colaboração, pois a organização pode optar pelo método de colaboração que melhor se adapte ao seu ambiente de trabalho. O *Enterprise Architect* também ganha destaque por ser uma ferramenta bastante utilizada por engenheiros de *softwares* e acadêmicos, isto porque trata-se de uma ferramenta que costuma corresponder as tendências do mercado, já que seu lançamento deu-se no ano de 2000 e desde então continua ativa nas organizações de desenvolvimento de *software* e instituições de ensino da engenharia de *software*.

Quando uma nova tecnologia é introduzida, ela passa por um ciclo de vida que nem sempre leva a uma aceitação ampla, apesar de as expectativas originais serem elevadas. “O grau segundo o qual qualquer tecnologia de engenharia de *software* ganha aceitação ampla está ligado a sua habilidade para resolver os problemas apresentados pelas tendências.” (PRESSMAN, 2011).

O sucesso e a vida útil de uma ferramenta estão de certa forma atrelados às tendências de mercado, em outras palavras, uma tendência de mercado pode justificar a escolha de uma ferramenta de apoio ao processo de desenvolvimento. Outro fator importante que pode acarretar no sucesso ou fracasso de uma organização na escolha de uma ferramenta é a capacidade e agilidade desta organização de extrair proveito dos processos e funcionalidades disponíveis na ferramenta, o que justifica a importância de estudos que evidenciam tais processos e funcionalidades a fim de auxiliar estas organizações a obterem maior proveito desta ferramenta.

Pressman (2011) declara como tendência tecnológica o desenvolvimento de sistemas baseado em modelos, onde modelos independentes de plataforma são criados em uma linguagem de alto nível, como UML(*Unified Modeling Language*), por exemplo, e então passam por vários níveis

Universidade do Estado de Santa Catarina
Centro de Educação Superior do Alto Vale do Itajaí

de transformação para eventualmente se transformarem em código-fonte para uma plataforma específica. Pressman (2011) conclui que o modelo e não mais o arquivo de código-fonte, deverá se tornar a nova unidade de saída. Um modelo facilitaria no aspecto de possuir muitas visões diferentes em diferentes níveis de abstração. No nível mais alto estariam as especificações de componentes independentes de plataforma; no nível mais baixo implementações específicas de cada plataforma.

Novotny (2004) apud Pressman (2011) afirma que uma nova geração de ferramentas funcionará em conjunto com um repositório para criar e armazenar modelos nos mais diversos níveis de abstração necessários, estabelecer relação entre os modelos e converter um modelo de um nível de abstração para outro, como por exemplo, converter um modelo de classes em código-fonte, gerenciar alterações e versões dos modelos e coordenar ações de controle e garantia de qualidade nos modelos de *software*. Com a crescente necessidade de trabalho colaborativo para atender as exigências do mercado, e as tendências tecnológicas do desenvolvimento de *software* apontando para os modelos de *software*, é natural que nos preocupemos como o desenvolvimento colaborativo destes modelos.

O objetivo geral do presente artigo é apresentar os métodos de desenvolvimento colaborativo da modelagem de *software* com a ferramenta CASE *Enterprise Architect* versão 11.1, edição corporativa, bem como auxiliar profissionais e estudantes da engenharia de *software* no processo de decisão para a escolha do método de colaboração que melhor se adapte ao ambiente onde este profissional ou estudante está inserido. Por fim apresentar os métodos de versionamento dos modelos e uma breve introdução sobre a segurança dos modelos na ferramenta *Enterprise Architect* versão 11.1, edição corporativa, para que se obtenha maior controle no desenvolvimento de modelos com o método de colaboração escolhido.

São cinco os objetivos específicos do presente artigo: (i) apresentar os diversos métodos de desenvolvimento colaborativo na ferramenta de modelagem de *software Enterprise Architect* versão 11.1, edição corporativa; (ii) analisar as vantagens e desvantagens de cada método de colaboração a fim de destacar os principais critérios que influenciam para a decisão de escolha de um método de colaboração; (iii) avaliar cada critério elencado em cada método de colaboração para possibilitar a geração de uma Matriz de Decisão predefinida para auxiliar e agilizar o processo de escolha de um método de colaboração; (iv) apresentar de forma breve os métodos de versionamento de modelos disponíveis na ferramenta CASE *Enterprise Architect* versão 11.1, edição corporativa, a fim de introduzir o profissional ou estudante de engenharia de *software* em um processo mais estável de desenvolvimento de modelos de *software*; (v) apresentar breve introdução sobre a segurança dos modelos e o controle de acesso na ferramenta CASE *Enterprise Architect* versão 11.1, edição corporativa.

A metodologia utilizada para a realização do presente artigo está dividida em 3 etapas: (i) pesquisar no manual do *Enterprise Architect* os diversos métodos de desenvolvimento colaborativo e versionamento de modelos desta ferramenta de modelagem de *software*, bem como as formas de segurança e controle de acesso; (ii) explorar de forma prática os métodos de colaboração do *Enterprise Architect* versão 11.1, edição corporativa, a fim de gerar opinião sobre as mesmas; (iii) analisar o resultado das pesquisas e da exploração do *Enterprise Architect* versão 11.1, edição corporativa, a fim de gerar insumos para a criação da Matriz de Decisão.

Este artigo está dividido nas seguintes seções: na próxima apresenta-se a revisão teórica e trabalhos correlatos; a terceira seção apresenta o trabalho colaborativo no *Enterprise Architect*, que está dividido em quatro subseções: (i) métodos de colaboração; (ii) escolha de um método de colaboração; (iii) métodos de versionamento; (iv) segurança; e na última seção as considerações finais desse artigo.

## 2. Revisão teórica e trabalhos correlatos

Universidade do Estado de Santa Catarina
Centro de Educação Superior do Alto Vale do Itajaí

O trabalho cooperativo ou colaborativo apoiado por computador é o principal pilar de apoio à presente pesquisa, para tanto, faz-se necessário ilustrar os conceitos pertinentes à esta metodologia de trabalho. Segundo Coll e Monereo(2008), o conceito de "trabalho cooperativo apoiado por computador", ou *Computer Supported Cooperative Word* (CSCW) tem sua origem em 1984, ano em que, respondendo a uma iniciativa da empresa *Digital Equipment Corporation* e ao *Massachusetts Instituteof Technology* (MIT), um grupo de desenvolvedores de *software* e pesquisadores de diversas áreas reuniram-se para explorar o papel da tecnologia em contextos profissionais de trabalho em grupo.

Segundo Cruz (2008), a ideia principal por trás do conceito CSCW, é de que as pessoas passem do trabalho individualizado, onde todos trabalham sem conhecer os processos nos quais suas atividades estão inseridas, para o trabalho em grupo, um trabalho efetivamente colaborativo, de forma que possam desempenhar suas responsabilidades sabendo "por que" fazem o que fazem, "para que" serve o produto que fazem e "para quem" irá o produto das suas atividades.

Segundo Fuks (2003) apud Rech (2007) o processo de colaboração consiste na troca de informações (comunicação) entre indivíduos de forma organizada (coordenação) operada em conjunto em um espaço compartilhado (cooperação). Segundo Cruz (2008), foi em decorrência do conceito CSCW que surgiram as ferramentas e componentes de auxílio à colaboração que formam o grande guarda-chuva chamado *groupware*. Segundo Baltzan e Phillips (2012), o *groupware* é um *software* que suporta a interação e a dinâmica da equipe. Ainda segundo Baltzan e Phillips (2012), o conceito de *groupware* integra vários sistemas e funcionalidades em um conjunto comum de serviços ou um aplicativo (cliente) único e pode representar uma grande variedade de sistemas e métodos de integração. Entende-se então que o objetivo principal dos *groupwares* é apoiar grupos de pessoas a realizarem tarefas em conjunto (colaborativamente), seja de forma simultânea ou não, a fim de alcançarem uma meta comum, mesmo trabalhando em localidades e fuso horários diferentes.

Segundo Pressman (2011), desde o princípio do projeto de *software*, todos os interessados devem compartilhar informações sobre metas, objetivos, desafios, requisitos, arquitetura, sobre praticamente todos os aspectos do *software* a ser criado, para que possa ocorrer uma colaboração aberta. Para que essa colaboração seja eficiente é necessário que utilizemos de *groupwares* como ferramentas de apoio, que vão desde simples ferramentas de *chat* até ferramentas complexas de modelagem de *software*, denominadas de ferramentas CASE, como é o caso da ferramenta *Enterprise Architect* versão 11.1, edição corporativa, objeto deste estudo.

Segundo Miyagusku (2008), uma ferramenta CASE consiste em um aplicativo que auxilia os profissionais da área de engenharia de *software* na tarefa de projetar e produzir *softwares*. Ainda segundo Miyagusku (2008), uma das principais características de uma ferramenta CASE é a modelagem visual, ou seja, a capacidade de representar através de modelos gráficos, o que está sendo definido e como isto está sendo feito. Segundo Pressman (2011), o objetivo de qualquer modelo é transmitir informações, desta forma, criam-se modelos para uma melhor compreensão do que será realmente construído.

Algumas ferramentas CASE podem possuir foco diferente de outras, para tanto, é conveniente mencionar a definição da ferramenta objeto deste estudo, *Enterprise Architect* versão 11.1, edição corporativa. A *Sparx Systems* (2014), empresa desenvolvedora da ferramenta *Enterprise Architect* 11.1, versão corporativa, lançada em agosto de 2014, define esta ferramenta como uma plataforma colaborativa de modelagem, design e gestão baseada em UML 2.4.1 e padrões relacionados. Uma grande solução empresarial para a visualização, análise, modelagem, testes e manutenção de uma ampla gama de sistemas, processos e arquiteturas.

Dentre os trabalhos correlatos, alguns se destacam por trazer grande contribuição para o presente artigo. Rech (2007) destaca as dificuldades de se trabalhar colaborativamente na etapa de modelagem de *software* devido as deficiências das ferramentas de modelagem existentes,

Universidade do Estado de Santa Catarina
Centro de Educação Superior do Alto Vale do Itajaí

propondo o desenvolvimento de uma ferramenta web para trabalho colaborativo na modelagem do *software*. O trabalho de Rech (2007) ajuda a compreender as dificuldades que desenvolvedores de *software* passam na etapa da modelagem, contribuindo para o foco de pesquisa do presente artigo a fim de detectar possíveis soluções para tais problemas.

Neto (2012) ressalta em seu trabalho o espaço que o desenvolvimento de *software* orientado à modelos vem assumindo na indústria e na academia e as dificuldades de se trabalhar colaborativamente neste modelo de desenvolvimento devido à complexidade das ferramentas e do ambiente que se necessita para tal, e então propõe um modelo de desenvolvimento auxiliado por *wiki* (site colaborativo), onde desenvolvedores sejam capazes de criar, compartilhar e versionar modelos, gerar código fonte e ainda documentar colaborativamente. Desta forma, o trabalho de Neto (2012) contribui para levantar questões tais como: será que as IDEs de modelagem de *software* estão preparadas para as tendências do mercado e o desenvolvimento colaborativo? É possível tornar o processo de desenvolvimento colaborativo de modelos um processo simples com o uso de uma IDE? A busca por respostas para estas e outras perguntas auxiliou na definição dos objetivos e serviu de motivação para a presente pesquisa.

No trabalho de Araújo (2011) é destacada a importância do trabalho colaborativo no desenvolvimento de *software* e as vantagens e desvantagens do uso de ferramentas que apoiem esta prática. São apresentadas também as principais características dos sistemas colaborativos, bem como exemplos de *softwares* que atuam nesta prática em diversos aspectos do desenvolvimento de *software*, tais como: comunicação, gerência de projeto, controle de erros e gestão de artefatos de documentação. O trabalho de Araújo (2011) fortalece o argumento de que o desenvolvimento colaborativo de *software* vem sofrendo ascendência e ganhando importância como fator de competitividade no mercado de *software*. Também enaltece a importância das ferramentas de apoio ao desenvolvimento colaborativo nesse processo evolutivo.

O principal diferencial entre o presente artigo e seus trabalhos correlatos é que o presente artigo tem por finalidade introduzir profissionais e estudantes de engenharia de *software* no desenvolvimento colaborativo de modelos de *software* apresentado métodos de colaboração e versionamento de modelos na ferramenta de modelagem de *software Enterprise Architect* 11.1, enquanto os demais trabalhos têm como linha geral destacar a importância e a dificuldade de se trabalhar colaborativamente no desenvolvimento de modelos de *software*.

## 3. O desenvolvimento colaborativo no *Enterprise Architect*

Os estudos e resultados apresentados neste artigo são baseados na versão 11.1, edição corporativa do *Enterprise Architect*, porém muitas das funcionalidades apresentadas também estão presentes em versões anteriores, como também em outras edições do *Enterprise Architect*. Para mais informações sobre versões e edições do *Enterprise Architect* consulte *Sparx System* (2014). A partir daqui usa-se a sigla EA para referir-se ao *Enterprise Architect* versão 11.1, edição corporativa.

Segundo a *Sparx System* (2014), o EA ajuda a criar projetos de desenvolvimento sob uma série de condições de trabalho, sendo que, algumas dessas condições se referem ao trabalho em equipes, seja em ambientes locais ou distribuídos, de forma síncrona ou assíncrona. Para que o trabalho colaborativo seja possível, algumas práticas são essenciais, tais como o compartilhamento do projeto, e outras práticas são recomendáveis, tais como o versionamento do projeto. O EA possui suporte à ambas as práticas e ainda dispõe de métodos flexíveis ao ambiente de trabalho.

### 3.1 Métodos de colaboração no *Enterprise Architect*

Universidade do Estado de Santa Catarina
Centro de Educação Superior do Alto Vale do Itajaí

O EA oferece um conjunto diversificado de funcionalidades projetadas especificamente para o compartilhamento de projetos entre os membros de uma equipe, são elas: diretório compartilhado; conexão com banco de dados; replicação e exportação/importação de XMI. As organizações possuem necessidades e ambientes de trabalho diferentes umas das outras, e cada organização procura por ferramentas e funcionalidades que venham de encontro às suas necessidades. Neste aspecto o EA se apresenta uma ferramenta poderosa por dispor desta flexibilidade para o compartilhamento de projetos.

### 3.1.1 Diretório compartilhado

A maneira mais fácil de compartilhar um projeto do EA entre um grupo de trabalho é colocar o arquivo do projeto (.eap) em uma unidade de rede compartilhada, para que estes profissionais possam se conectar a este a partir de suas estações de trabalho. Desta forma estes profissionais podem realizar alterações no projeto de forma concomitante.

Imaginemos um cenário com três profissionais trabalhando de forma colaborativa no desenvolvimento de modelos de *software* com o auxílio do método de colaboração de diretório compartilhado, conforme apresenta a Figura 1. As atualizações realizadas no projeto por um usuário X não são visíveis pelos demais usuários de forma automática. Para que estas atualizações sejam visíveis aos demais usuários é necessário que o usuário X salve as alterações realizadas. Neste ponto as alterações realizadas pelo usuário X já estão disponíveis, porém ainda não visíveis aos demais usuários. Para que fiquem visíveis aos usuários demais usuários, estes devem realizar o comando de atualização do projeto no menu *File ->Reload Project*. Se os usuários X e Y e Z estiverem colaborando no mesmo diagrama ao mesmo tempo, assim que um dos usuários salvar suas alterações os outros ficarão bloqueados para alterações neste diagrama até que executem o comando de atualização do projeto.

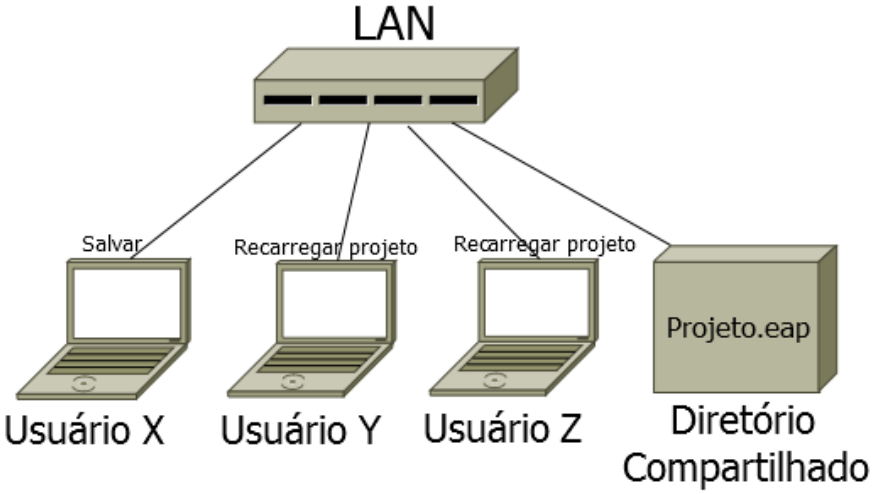



**Figura 1 – Diretório compartilhado (produção do próprio autor, 2014)**

Apesar da utilização de um diretório compartilhado ser bastante simples, este método de colaboração possui um grau de instabilidade que pode acarretar em inconsistências que tornam o projeto complexo de ser gerido. Esta instabilidade pode ser acarretada por alguns fatores que podem gerar resultados inesperados ou até mesmo corromper o arquivo do projeto, são eles: falha da rede; falha no computador de algum dos usuários conectados ao projeto; dois ou mais usuários modificando o mesmo diagrama; falha de gravação física do arquivo.

O EA reconhece que estas falhas podem ocorrer e dispõe de uma ferramenta para reparar projetos corrompidos, que se encontra no menu *Tools -> Data Management ->Manage .EAP*

Universidade do Estado de Santa Catarina
Centro de Educação Superior do Alto Vale do Itajaí

*File ->Repair .EAP File*, porém esta prática não garante que 100% das modificações realizadas sejam recuperadas.

As principais vantagens deste método de colaboração são: simplicidade de implantação; facilidade de aprendizado do processo; dispensa integração com outras ferramentas; possibilita a colaboração de forma síncrona. As principais desvantagens são: dependência da velocidade da rede; possibilidade de perda de modificações; não possui ferramenta para auxiliar o *merge*(fusão)de alterações conflitantes; inadequado para grandes equipes; não possibilita o trabalho distribuído, ou seja, os colaboradores tem que estar todos conectados à uma rede local.

### 3.1.2 Conexão com banco de dados

Com a finalidade de dar suporte à um número maior de usuários simultâneos o EA também possibilita o uso de um Sistema Gerenciador de Banco de Dados (SGBD) como repositório dos dados do projeto. A versão atual 11.1 do EA fornece suporte aos seguintes SGBDs: SQL Server 2000, 2005 e 2008; SQL Server Express 2005 e 2008; MySQL 4 e 5; PostgreSQL 7, 8 e 9; Adaptive Server Anywhere 8 e 9; SQL Anywhere 10, 11 e 12; Access 2007; ProgressOpenEdge; MSDE e Oracle 9i, 10g e 11g.

Neste método de colaboração é possível tanto converter um projeto já existente (.eap) para um repositório SGBD, como converter um projeto SGBD para um arquivo .eap, o que fornece flexibilidade ao usuário na troca de repositórios, que pode decorrer de um crescimento da equipe ou de uma mudança na estrutura interna, por exemplo. Uma das grandes vantagens deste método de colaboração é que os usuários não precisam estar alocados no mesmo espaço físico, já que através da conexão ODBC (*Open Database Connectivity*) é possível manipular o projeto de qualquer lugar desde que haja uma conexão com a internet, conforme apresenta a Figura 2.

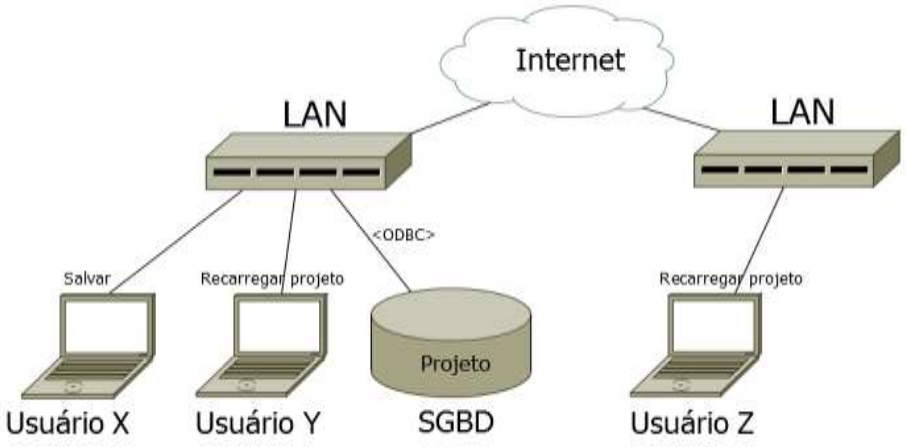



**Figura 2 – Conexão com banco de dados (produção do próprio autor, 2014)**

Outra característica importante deste método de colaboração é a segurança fornecida pelo controle de transações dos bancos de dados relacionais, o que torna este método de colaboração muito mais estável que o método do diretório compartilhado. Da mesma forma que o diretório compartilhado, para visualizar as alterações realizadas por outros usuários é necessário invocar o comando de atualização no menu *File ->Reload Project*.

Apesar de não ser tão simples de se configurar quanto um diretório compartilhado, a configuração de um SGBD como repositório do projeto não é algo complexo. Este processo requer a execução de seis etapas, são elas: (i) instalação de um SGBD suportado pelo EA; (ii) criação de um repositório no SGBD instalado e execução do script (.sql) disponibilizado pela *Sparx Systems* (2014) que irá gerar a estrutura de tabelas necessárias para armazenamento do

Universidade do Estado de Santa Catarina
Centro de Educação Superior do Alto Vale do Itajaí

projeto (o arquivo do projeto é convertido em uma estrutura de dados relacional que é armazenado no SGBD); (iii) instalação do driver ODBC correspondente ao SGBD instalado; (iv) configuração da conexão entre o EA e o SGBD; (v) verificação da integridade do projeto. O EA dispõe de uma funcionalidade localizada no menu *Tools -> Data Management -> Project Integrity Check*, para verificar a integridade do projeto, pois quedas de rede, energia ou ainda outros fatores podem causar inconsistências no projeto, tais como elementos órfãos, por exemplo. Este processo de verificação da integridade do projeto corrige essas inconsistências indesejadas que podem acarretar em problemas futuros. (vi) transferência do arquivo .eap para o repositório SGBD através do menu *Tools -> Data Management -> Project Transfer*.

As principais vantagens deste método de colaboração são: suporte a grande número de usuários simultâneos; possibilita o trabalho geograficamente distribuído quando conectado à internet; segurança transacional; possibilita a colaboração de forma síncrona. As principais desvantagens são: dependência da velocidade da rede; não possui ferramenta para auxiliar o *merge* de alterações conflitantes; necessidade de conhecimento básico de administração de algum SGBD suportado pelo EA.

### 3.1.3 Replicação

Além de compartilhar o projeto em tempo real através de uma rede, o EA também permite o trabalho colaborativo *offline*, ou seja, independente de uma estrutura de rede ou de conexão com a internet. Isto é possível através da replicação do projeto. Este método de colaboração consiste em gerar réplicas físicas do arquivo (.eap) do projeto, possibilitando que os vários colaboradores trabalhem em arquivos independentes e de tempos em tempos realizem a sincronização dessas réplicas à fim de gerar um arquivo único do projeto contento todas as alterações realizadas pelos diversos colaboradores, conforme apresenta a Figura 3, onde o Usuário X é o gerador das réplicas e também o responsável por sincronizar essas réplicas ao fim de cada ciclo de alteração, gerando um arquivo único com todas as alterações realizadas pelos usuários X, Y e Z, e então novas réplicas podem ser geradas para a continuidade do processo de colaboração.

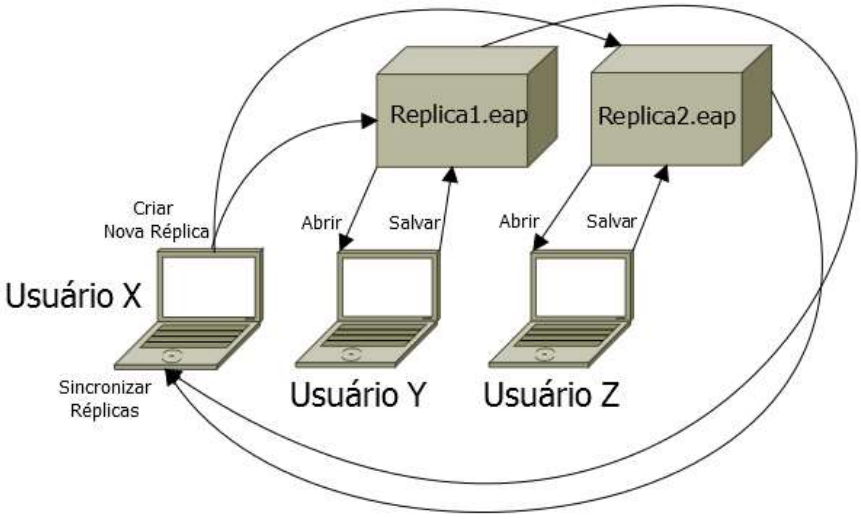



**Figura 3 – Replicação (produção do próprio autor, 2014)**

Apesar de parecer simples, a gestão deste método de colaboração é um tanto quanto complexa e é preciso estar atento a duas regras básicas para evitar que se perca o controle das alterações, que pode resultar em inconsistências como perdas ou redundâncias de informações. As regras são: (i) a adição de artefatos (diagramas, classes, ligações, etc.) é acumulativa, ou seja, se três réplicas receberem dois novos diagramas cada, após a sincronização haverá seis diagramas; (ii)

Universidade do Estado de Santa Catarina
Centro de Educação Superior do Alto Vale do Itajaí

exclusões sempre prevalecem sobre alterações, ou seja, se uma classe for alterada em uma das réplicas e excluída em outra, após a sincronização a classe deixa de existir.

Neste método de colaboração também não se aconselha que dois ou mais usuários trabalhem no mesmo pacote, pois podem ocorrer conflitos, porém, se for necessário, o EA conta com uma ferramenta de auxílio à resolução de conflitos para este método de colaboração, mas para que a resolução desses conflitos seja bem-sucedida, é importante que os usuários que realizaram as alterações acompanhem o processo de sincronização para que nenhuma alteração seja perdida.

As principais vantagens deste método de colaboração são: possibilita o trabalho colaborativo geograficamente distribuído de forma *offline*; dispensa integração com outras ferramentas; o EA disponibiliza funcionalidade para auxílio à resolução de conflitos. As principais desvantagens são: as alterações realizadas por outros membros da equipe não são visíveis em tempo real, o que se torna uma grande desvantagem quando há uma maior dependência entre os artefatos gerados por diferentes colaboradores; é necessário tempo e cuidado para realizar a sincronização das réplicas.

### 3.1.4 Exportação/Importação de XMI

Segundo a *Sparx Systems* (2014), *XML Metadata Interchange* (XMI) definido pela OMG (*Object Modeling Group*) é um formato padrão aberto baseado em XML que permite o intercâmbio de informações de modelos entre diversas ferramentas que aplicam este padrão. Ainda segundo a *Sparx Systems* (2014), o EA utiliza XMI como um método de importação e exportação de especificações de modelos entre diferentes projetos e pacotes UML.

Assim como a replicação, este método de colaboração também permite o trabalho colaborativo de forma *offline*, já que a junção do trabalho dos diversos membros da equipe é realizado de forma assíncrona com a exportação e importação de arquivos no formato XMI. O processo de colaboração é bastante semelhante ao método da replicação, como apresenta a Figura 4, porém existe um importante diferencial entre o método de exportação/importação de XMI e a replicação, que é a possibilidade de tratar pacotes do projeto de forma individualizada, ou seja, o EA permite exportar/importar pacotes, o que possibilita a reutilização desses pacotes em diversos projetos, por exemplo, imagine que temos alguns pacotes que são padrões em todos os projetos da organização. Desta forma podemos manter esses pacotes padrões em um único projeto que poderíamos chamar de “Projeto Padrão”, onde que, todos os pacotes mantidos no “Projeto Padrão” serão salvos em um diretório compartilhado ou em um repositório de controle de versão, e então esses pacotes estariam disponíveis para serem importados nos demais projetos.

REAVI Revista Científica do Alto Vale do Itajaí
Universidade do Estado de Santa Catarina
Centro de Educação Superior do Alto Vale do Itajaí

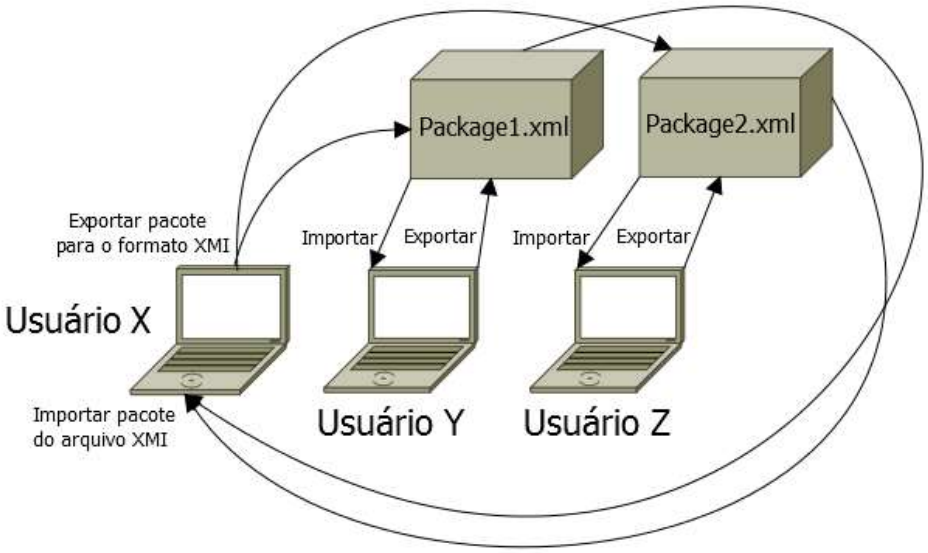



**Figura 4 – Exportação/Importação XMI (produção do próprio autor, 2014)**

Este método de colaboração permite a comparação entre versões de um mesmo pacote, sem a necessidade de comparar o projeto inteiro, porém é aconselhável que os colaboradores não trabalhem no mesmo pacote, uma vez que a importação de um pacote sobrescreve todos os dados do seu local de destino. Deve-se evitar também qualquer dependência entre os pacotes, pois se um pacote for importado para um projeto onde o pacote dependente não exista, esta dependência (ligação) deixa de existir e quando este pacote for importado novamente para seu local de origem perde-se esta dependência.

As principais vantagens deste método de colaboração são: permite a reutilização de pacotes entre diversos projetos; possibilita o trabalho colaborativo geograficamente distribuído de forma *offline*; dispensa integração com outras ferramentas; possibilita a comparação entre versões de um mesmo pacote; possibilita o intercâmbio de informações com outras ferramentas que aplicam o padrão XMI. As principais desvantagens são: alterações realizadas por outros membros da equipe não são visíveis em tempo real; dificuldade de gestão de dependências entre pacotes; informações podem ser sobrescritas sem a possibilidade de recuperação; inadequado para equipes que desejam colaborar em um mesmo pacote pelo fato de não fazer merge das alterações e sim substituição pelo pacote importado.

## 3.2 Optando por um método de colaboração

Agora já sabemos que o EA conta com quatro métodos de colaboração no desenvolvimento de modelos, cada qual com suas características e qualidades, são eles: (i) diretório compartilhado; (ii) conexão com banco de dados; (iii) replicação; (iv) exportação/importação de XMI. Para auxiliar na escolha de um método de colaboração, este artigo propõe a utilização de uma Matriz de Decisão. Segundo César (2013) utiliza-se uma Matriz de Decisão quando temos uma resposta afirmativa para as seguintes questões: (i) existe uma lista de opções para um determinado objetivo?; (ii) é necessário selecionar uma opção ou priorizar as opções disponíveis?; (iii) a melhor opção não é óbvia?

Ainda Segundo César (2013), uma Matriz de Decisão ou Matriz de Priorização como também é chamada, é uma ferramenta de auxílio à tomadas de decisão, ela utiliza de uma metodologia lógica para delimitar o foco quando se tem um grande número de opções de escolha ou de critérios que influenciam nesta escolha e a melhor opção não é óbvia.

Universidade do Estado de Santa Catarina
Centro de Educação Superior do Alto Vale do Itajaí

Segundo Norton (2009), a Matriz de Decisão ocasionalmente ajuda você a identificar a melhor opção, forçando você a considerar os vários critérios de decisão de forma sistemática. A Matriz de Decisão tem formato de tabela, e pode ser apresentada de forma diferente por diferentes autores, mas sempre com o mesmo objetivo. A Matriz de Decisão para o método de colaboração é apresentada na Tabela 1, onde são listados nas linhas os critérios de decisão e nas colunas as opções de escolha. Então é definido um fator de ponderação ou peso, como também é chamado, para cada critério de decisão de acordo com a sua importância. Colaboração em tempo real, por exemplo, pode ser mais importante para a equipe do que a possibilidade de colaboração *offline* ou vice-versa. A escolha do intervalo do fator de ponderação é subjetiva e escolhida pelo desenvolvedor. Para facilitar a definição do fator de ponderação de cada critério de decisão utilizaremos na nossa Matriz de Decisão o intervalo de 0 à 5, que correspondem respectivamente à: não se aplica; dispensável; pouco importante; importante; muito importante; indispensável. Na nossa Matriz de Decisão representada na Tabela 1, o fator de ponderação está preenchido de forma aleatória, apenas com o intuito de demonstrar como o cálculo é efetuado, mas ele deve ser definido de acordo com o que julgar-se mais importante dentre os critérios apresentados.

Na Matriz de Decisão também deve ser definida uma nota para cada opção de escolha, neste caso, cada método de colaboração deve ser avaliado dentro de cada critério de decisão. Na Matriz de Decisão representada na Tabela 1 esta avaliação foi realizada pela presente pesquisa através da consulta do Manual do Usuário do *Enterprise Architect* e da exploração prática da ferramenta a fim de poupar seu tempo e auxiliar você nesta tomada de decisão. Para a definição das notas foi utilizado o intervalo de 0 a 5, que correspondem respectivamente à: não atende; péssimo; ruim; regular; bom; ótimo.

Definidos os fatores de ponderação e as notas, basta multiplicar o fator de ponderação de cada critério de decisão pela nota atribuída à este critério dentro de cada opção de escolha, gerando uma pontuação de cada critério de decisão para cada opção de escolha. Somam-se então as pontuações dos critérios de decisão de cada opção de escolha gerando uma pontuação total da opção de escolha. A opção que obtiver a maior pontuação tende a ser a melhor escolha de acordo com os critérios avaliados.

Na Matriz de Decisão representada na Tabela 1 são apresentadas nas colunas as quatro opções de escolha que temos para o trabalho colaborativo no EA, e nas linhas os critérios de decisão que se julga serem os mais influentes na escolha do método de colaboração, conforme as vantagens e desvantagens de cada método de colaboração.

No cenário fictício criado para demonstração do resultado da Matriz de Decisão representada na Tabela 1, está sendo priorizada a colaboração em tempo real, usabilidade em pequenas equipes, garantia de integridade e auxílio a resolução de conflitos. Para este cenário em questão a melhor opção apontada pela matriz é o método de colaboração por banco de dados, pois alcançou maior pontuação na soma das pontuações dos critérios de decisão, conforme apresentado na linha do total na Tabela 1, onde temos o valor 98 grifado em vermelho. A soma das pontuações deste método de colaboração foi maior porque dentre os critérios que foram dados como prioritários, ou seja, critérios em que o fator de ponderação é igual a muito importante ou indispensável, este método de colaboração obteve maior nota na avaliação dos mesmos, por exemplo, o critério "garantia de integridade dos modelos" foi definido como um fator indispensável (fator 5, na escala de 0 a 5), e a nota dada a este critério de decisão no método de colaboração por banco de dados foi "bom" (nota 4, na escala de 0 a 5), fazendo com que este método obtivesse a pontuação 20 (fator de ponderação multiplicado pela nota), que foi a maior pontuação obtida dentre os métodos de colaboração neste critério de decisão.

De acordo com a classificação definida pela pontuação total, a segunda melhor opção seria o método de colaboração por diretório compartilhado, que atingiu a pontuação 86, em terceiro o método de replicação com a pontuação 75 e por último o método de exportação/importação de

REAVI
Revista Científica do Alto Vale do Itajaí
Universidade do Estado de Santa Catarina
Centro de Educação Superior do Alto Vale do Itajaí

XMI com a pontuação 59 conforme apresenta a Tabela 1. É possível também que outros critérios de decisão sejam adicionados à tabela se forem necessários, na Tabela 1 foram descriminados apenas os critérios de decisão avaliados em cada método de colaboração pelo presente artigo, isto não impede de que o próprio desenvolvedor adicione um novo critério e o avalie dentro de cada método de colaboração, atribuindo notas para cada um dos métodos.

Norton (2009) sugere precaução na aplicação dos resultados da Matriz de Decisão devido a subjetividade na definição dos fatores de ponderação e das notas e declara que o valor real da Matriz de Decisão está em dividir o problema em elementos que podem ser facilmente resolvidos, nos forçando a pensar no valor relativo de cada opção de escolha em vários critérios de decisão e então tomar uma decisão com maior confiança sobre qual é a melhor opção.

| Critério de decisão | Fator de ponderação | Diretório compartilhado | | Banco de dados | | Replicação | | Exportação/importação de XMI | |
|---|---|---|---|---|---|---|---|---|---|
| | | Nota | Pontuação | Nota | Pontuação | Nota | Pontuação | Nota | Pontuação |
| Colaboração *offline* | 0 | 0 | 0 | 0 | 0 | 4 | 0 | 3 | 0 |
| Colaboração em tempo real | 4 | 4 | 16 | 5 | 20 | 0 | 0 | 0 | 0 |
| Colaboração geograficamente distribuída | 1 | 0 | 0 | 4 | 4 | 3 | 3 | 3 | 3 |
| Usabilidade em grandes equipes (Mais de 3 colaboradores) | 2 | 3 | 6 | 4 | 8 | 2 | 4 | 2 | 4 |
| Usabilidade em pequenas equipes (Até 3 colaboradores) | 4 | 5 | 20 | 5 | 20 | 3 | 12 | 3 | 12 |
| Usabilidade na colaboração no mesmo pacote | 1 | 2 | 2 | 3 | 3 | 2 | 2 | 1 | 1 |
| Garantia de integridade dos modelos | 5 | 2 | 10 | 4 | 20 | 3 | 15 | 3 | 15 |
| Auxílio à resolução de conflitos | 5 | 2 | 10 | 2 | 10 | 5 | 25 | 2 | 10 |
| Simplicidade da implantação | 2 | 5 | 10 | 2 | 4 | 4 | 8 | 4 | 8 |
| Simplicidade da gestão | 3 | 4 | 12 | 3 | 9 | 2 | 6 | 2 | 6 |
| **Total** | | | **86** | | **98** | | **75** | | **59** |

**Tabela 1 – Matriz de Decisão (produção do próprio autor, 2014)**

Seja qual for a melhor opção apontada pela Matriz de Decisão, o mais importante é que se de inicio a colaboração para que seja possível que o próprio desenvolvedor sinta se o método atende ou não as suas necessidades, uma vez que o EA permite a migração a qualquer momento entre um método de colaboração e outro, fazendo com que o desenvolvedor não seja refém da sua escolha, e sim que esta escolha seja uma boa porta de entrada à colaboração no desenvolvimento de modelos de *software*.

É importante também que indiferente do método escolhido seja realizado o controle de versão dos modelos para que se possa manter a integridade dos mesmos e obter maior segurança nas alterações, pois o desenvolvedor não precisará "temer" a exclusão de artefatos que considera obsoleto. Segundo Dornbusch, Fischer e Startz (2014), a liberdade de excluir é um grande avanço, pois livrar-se de ideias e implementações velhas permite que o time experimente coisas novas e melhore o produto final.

## 3.3 Métodos de versionamento no *Enterprise Architect*

O termo "versionamento" é bastante comum para quem trabalha no desenvolvimento de código fonte de *softwares*, porém não é muito conhecido pelos profissionais que desenvolvem modelos e outros artefatos de documentação de *software*. Acredita-se que um dos principais fatores que justifica esta afirmação está relacionado com carência de recursos de controle de versão das ferramentas CASE, porém este patamar está mudando, as ferramentas CASE estão se

Universidade do Estado de Santa Catarina
Centro de Educação Superior do Alto Vale do Itajaí

modernizando. Um exemplo disto é o EA versão 11.1 que conta com vários recursos no âmbito de versionamento de modelos.

> Uma consequência de ter cada versão de cada arquivo sob controle de versão é que isso permite que você seja agressivo em relação a excluir coisas que acha que não irá precisar. Com controle de versão, você pode responder à questão "Devermos excluir esse arquivo?" com um "Sim!" sem risco; se for a decisão errada, é simples corrigi-la recuperando uma versão anterior. (DORNBUSCH; FISCHER; STARTZ, 2014)

Apresenta-se à seguir alguns destes recursos sob dois métodos distintos de controle de versão, cada qual com suas vantagens e desvantagens, são eles: (i) pacotes controlados; (ii) Ferramentas de controle de versões.

### 3.3.1 Pacotes controlados

Pacote controlado é o termo utilizado pelo EA para designar um pacote que esteja sob o controle de versões interno do EA. A principal virtude de um pacote controlado é a possibilidade de criar *baselines* (pontos de restauração do projeto) deste pacote como uma forma de controle manual de versões, possibilitando realizar comparações do projeto atual com as *baselines* anteriores, como também recuperar versões anteriores. O EA permite visualizar todas as alterações realizadas em comparação com um *baseline* na forma de texto, possibilitando selecionar quais alterações devem ser efetivas e quais devem retornar ao estado do *baseline*, conforme apresentado na Figura 5.

A Figura 5 é uma representação da comparação do projeto atual com um *baseline* anterior, onde podemos notar que a Classe 2 em destaque sofreu uma alteração de posicionamento nas propriedades *Top* (distância do elemento em relação ao topo do diagrama) e *Bottom* (distância do elemento em relação a margem inferior do diagrama) na comparação entre as colunas *Model* (projeto atual) e *Baseline* (versão anterior do projeto), o que indica que o Classe 2 sofreu um deslocamento para baixo conforme demonstra graficamente a Figura 6.

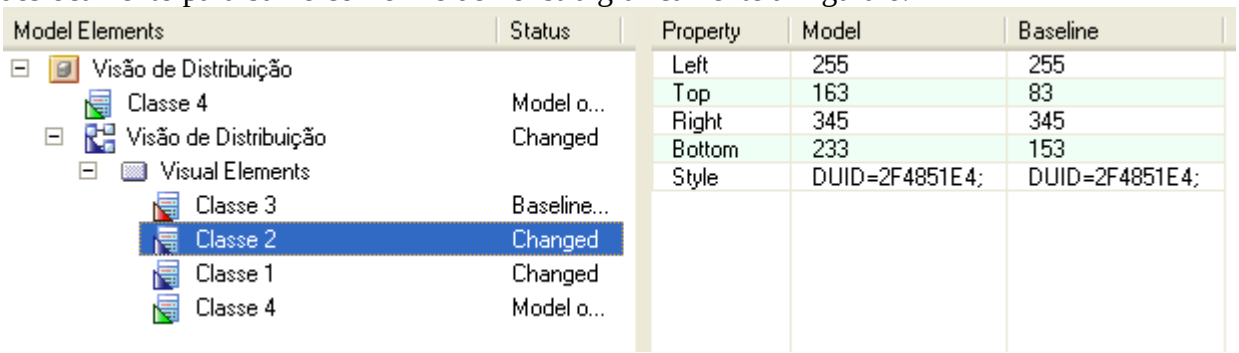
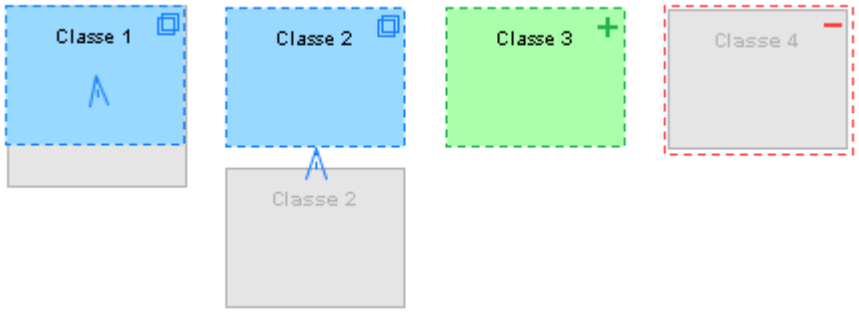



**Figura 5 – Comparação textual com *baseline*(produção do próprio autor, 2014)**

Há também a possibilidade de comparar as alterações de um determinado diagrama com um *baseline* de forma gráfica, mas apenas para alterações de *layout*, tais como: inclusões, exclusões, redimensionamentos e deslocamentos de objetos visuais, conforme apresentado na Figura 6, onde a Classe 1 foi redimensionada, a Classe 2 deslocada, a Classe 3 foi excluída e a Classe 4 adicionada. Após selecionadas as alterações o EA indica a ação que irá efetuar (conforme apresenta a coluna "*Action*" da Figura 6) se o usuário confirmar que deseja reverter as alterações para o *baseline* previamente selecionado.




| Name | Status | Action |
|---|---|---|
| Classe 3 | Baseline only | Restore Element & add to Diagram |
| Classe 2 | Changed | Restore Element position |
| Classe 1 | Changed | Restore Element position |
| Classe 4 | Model only | Delete from Diagram |

**Figura 6 - Comparação gráfica com *baseline*(produção do próprio autor, 2014)**

Para mais informações sobre Pacotes Controlados consulte a documentação do EA em *Sparx System* (2014).

### 3.3.2 Ferramentas de controle de versão

Com a utilização de uma ferramenta de controle de versões aliada à utilização de pacotes controlados, possuímos uma série de benefícios, dentre os quais se destacam: manter o histórico das alterações efetuadas pelos vários membros da equipe; capacidade de recuperar revisões anteriores; propagação de atualização de modelos entre os diversos colaboradores da equipe; coordenação da partilha de pacotes entre os colaboradores da equipe.

O controle de versão pode ser realizado de duas formas: (i) internamente, ou seja, através de ferramentas de controle de versão suportadas pelo EA, são elas: Subversion, CVS, SCC e TFS. Desta forma comandos como *check-in* e *check-out* são habilitados dentro da ferramenta, proporcionando maior integração entre o EA e o controle de versões, visando a facilidade e agilidade do processo; (ii) de forma manual, exportando os pacotes no formato XMI para um diretório que esteja submetido ao controle de versões de sua preferência, e então o controle de *check-in/check-out* é realizado externo ao EA, o que pode onerar o processo, mas também se torna uma opção quando a organização já possui uma ferramenta de controle de versões (que não é suportada internamente pelo EA) e não deseja adotar outra apenas para o processo de versionamento dos artefatos de modelagem.

Além do versionamento dos modelos devemos nos atentar a segurança dos mesmos, pois com vários desenvolvedores alterando o projeto alterações inadvertidas podem ocorrer, comprometendo a integridade do projeto.

## 3.4 Segurança

Segundo a *Sparx Systems* (2014), a segurança no EA é um meio de melhorar a colaboração no desenvolvimento de modelos, impedindo, por exemplo, que inadvertidamente sejam realizadas alterações em elementos (pacotes, modelos ou objetos) por usuários que não foram designados como autores destes elementos. Ainda segundo a *Sparx System* (2014) o EA conta com duas políticas de segurança: (i) modo de segurança padrão, onde inicialmente todos os elementos estão liberados para edição e se necessário o administrador pode bloquear o acesso a um elemento ou um grupo de elementos para um usuário ou para um grupo de usuários; (ii) modo de

Universidade do Estado de Santa Catarina
Centro de Educação Superior do Alto Vale do Itajaí

segurança rigoroso, é a inversão da primeira política, neste modo todos os elementos possuem acesso apenas para a leitura e somente são liberados para edição se o administrar explicitar um usuário ou grupo de usuários para acessar estes elementos.

Desta forma o papel de um administrador de segurança no EA é primeiramente definir qual política de segurança será utilizada para o trabalho colaborativo e posteriormente permitir ou restringir acesso à elementos para usuários ou grupos de usuários. Usuários comuns também podem realizar a manutenção da segurança de seus elementos, dentre as funcionalidades disponíveis à um usuário comum, destacam-se: bloquear elementos e visualizar quais usuários ou grupos de usuários estão bloqueados para alteração à um determinado elemento. Para saber como habilitar uma política de segurança e gerir grupos e usuários consulte *Sparx Systems* (2014).

## 4. Considerações Finais

Apesar de não ser uma novidade, o trabalho colaborativo no desenvolvimento de modelos de *software* ainda é um paradigma, pois há pouco material de apoio à esta metodologia de trabalho, como ferramentas, descrições de processos e relatos de experiências, o que torna esta metodologia de certa forma imatura.

O *Enterprise Architect*, versão 11.1, edição corporativa, se mostrou uma ferramenta bastante flexível no desenvolvimento colaborativo de modelos de *software*, permitindo várias formas de colaboração que atendem aos mais variados ambientes de desenvolvimento, porém a complexidade dos modelos UML em comparação com arquivos de código fonte, faz com que ainda estejamos um pouco distantes da naturalidade da colaboração no desenvolvimento de códigos fonte de *softwares*.

Não existe um método de colaboração ideal, a escolha do método está relacionada ao ambiente de projetoe às necessidades da organização. Indiferente qual seja o método escolhido, é importante que se trabalhe em conjunto com um método de controle de versão a fim de se adquirir consistência e organização no desenvolvimento dos modelos.

Para se obter os melhores resultados possíveis, não basta apenas escolher o método de colaboração mais adequado ou o método de versionamento mais robusto, é necessário que haja um processo bem definido que organize o grupo, defina os papéis e responsabilidades.

Apesar de possuir grande suporte a colaboração com métodos de colaboração, versionamento, controle de acesso e outras funcionalidades que fogem ao escopo do presente artigo, o*Enterprise Architect* ainda precisa evoluir bastante no aspecto da colaboração concorrente, pois o processo de modelagem é um processo onde decisões importantes são tomadas e a colaboração concorrente tende à beneficiar muito este processo.

Como sugestão à trabalhos futuros, pode-se vislumbrar a colaboração como auxílio no desenvolvimento orientado à modelos, a comparação entre ferramentas que auxiliam no processo de colaboração e estudos de caso que evidenciem os benefícios e limitações da colaboração no desenvolvimento de modelos de *software*.

## 5. Referências

ARAÚJO, Marco Antônio Pereira. **Desenvolvimento de *software* apoiado *groupware*. Revista Engenharia de *Software***. n. 34, 2012. Disponível em: <http://www.devmedia.com.br/websys.5/webreader.asp?cat=48&artigo=3361&revista=esmagazine_34#a-3361>. Acesso em: 9 setembro 2014.

BALTZAN, Paige; PHILLIPS, Amy. **Sistemas de informação**. Porto Alegre: AMGH, 2012.

Universidade do Estado de Santa Catarina
Centro de Educação Superior do Alto Vale do Itajaí

CÉSAR, Francisco I. Giocondo. **Ferramentas gerenciais da qualidade**. São Paulo: Biblioteca24horas, 2013.

COLL, César; MONEREO, Carles. **Psicologia da educação virtual:** aprender e ensinar com as tecnologias da informação e da comunicação. Porto Alegre: Artmed, 2008.

CRUZ, Tadeu. **BPM & BPMS**: *business process management & business management systems*. Rio de Janeiro: Brasport, 2008.

DORNBUSCH, Rudiger; FISCHER, Stanley; STARTZ, Richard. **Entrega contínua:** como entregar *software*. Porto Alegre: Bookman, 2014.

FUKS, Hugo. **Sistemas colaborativos**. Rio de Janeiro: Elsevier Editora Ltda, 2012.

MARTINAZZO, Alexandre AntonioGolçalves. **Considerações sobre desenvolvimento colaborativo de *software* para aprendizagem em plataformas móveis**. 2011. Dissertação de mestrado, Mestrado em Engenharia, Universidade de São Paulo, São Paulo, 2011.

MIYAGUSKU, Renata. **Informática para concursos públicos**. São Paulo: Digerati Books, 2008.

NETO, David Fernandes. **CoMDD:** uma abordagem colaborativa para auxiliar o desenvolvimento orientado à modelos. 2012. Dissertação de mestrado, Ciências de Computação e Matemática Computacional, Universidade de São Paulo, São Paulo, 2012.

NORTON, Robert L.. **Cinemática e dinâmica dos mecanismos**. São Paulo: Bookman, 2009.

PRESSMAN, Roger S. **Engenharia de *Software*:** uma abordagem profissional. 7. ed. Porto Alegre: AMGH, 2011.

RECH, Vagner. **Ferramenta web colaborativa para modelagem de casos de uso**. 2007. Trabalho de conclusão do curso de Ciências da Computação, Universidade do Vale do Itajaí, Itajaí, 2007.

*SPARX SYSTEMS*.***Enterprise Architect User Guide***.2014. Disponível em: <http://www.sparxsystems.com.au/bin/EAUserGuide.pdf>. Acesso em: 13 agosto 2014.

TEIXEIRA, Tiago Mourão. ***Web collaboration for software engineering***. 2009. Dissertação de mestrado, Engenharia Informática e Computação, Faculdade de Engenharia, Universidade do Porto, 2009.