ADEQUAÇÃO DO PROCESSO UNIFICADO PARA DESENVOLVIMENTO DE SISTEMAS GRP (*GOVERNMENT RESOURCE PLANNING*)

Mauro Borges França[1,2], Alexandre Cardoso[2] e Edgard A. Lamounier Jr.[2]
[1]Instituto Federal de Educação, Ciência e Tecnologia do Triângulo Mineiro (IFTM) Uberaba, MG – Brasil
[2]Universidade Federal de Uberlândia (UFU) - Uberlândia, MG – Brasil
mauro@iftm.edu.br, {alexandre,edgard}@ufu.br

***Resumo* - Obter um processo de software adequado ao desenvolvimento confiável, de boa aderência e que não dispenda tempo demasiado na geração de artefatos é forte motivação para responder demandas de mercado, e portanto, requer pesquisa. Este trabalho propõe a adequação de metodologias tradicionais, relacionados com processos unificados, visando adequar o desenvolvimento de software característico de pequenas e médias empresas de tecnologia. Como estudo de caso, o desenvolvimento de um sistema de gestão pública é considerado.**

***Palavras-Chave* -: Processo de Software, metodologias ágeis, metodologias tradicionais, GRP, produtividade.**

ADEQUACY OF THE UNIFIED PROCESS FOR DEVELOPING GRP SYSTEMS (GOVERNMENT RESOURCE PLANNING)

***Abstract* - Getting a proper trustful process development, good, robust and not too long to use in the generation of artifacts is a strong motivation to answer market demands and, therefore requites quality research. This paper proposes the adaptation of traditional methods in order to accommodate some conformity, aiming at the development process characteristic of small or medium sized software factories. As a case study, the techniques proposed here are applied to a GRP system for public institutions.**

***Keywords* – Process Software, Agile, traditional methodologies, GRP, productivity.**

I. INTRODUÇÃO

A Engenharia de Software nasceu da necessidade de desenvolver, por métodos de engenharia, a produção de programas de computadores, envolvendo processos e métricas de tempo, custo, envolvimento de pessoal, resultados e possíveis avanços.

Considerando a Engenharia de Software, é notório dizer que desenvolver software de qualidade é um grande desafio para as fábricas de software, independente de seu porte e encontrar o processo adequado ao desenvolvimento é fator de estudos por vários pesquisadores no mundo. O trabalho em [11] relata que o surgimento de padrões internacionais para processos de software, como a Norma ISO/IEC 12207 e os modelos de maturidade (CMM, TRILLIUM, BOOTSTRAP e ISO/IEC 15504), influenciaram as organizações a direcionar seus esforços não só na definição de processos, mas também, no estabelecimento de mecanismos para melhorá-los, continuamente. No entanto, o desenvolvimento de software continua, em muitos casos, sendo uma atividade fortemente dependente das habilidades individuais dos desenvolvedores, haja vista que em várias organizações ainda não existem processos definidos e pouco conhecimento quanto à maturidade de processos.

No âmbito dos processos de software, ressalta-se que as fabricas de software de qualquer organização, utilizam algum processo de desenvolvimento de software, seja em pequenos, médios ou grandes projetos, mesmo que sejam processos informais. A adoção de metodologias de desenvolvimento orientam os integrantes do projeto para as atividades, ações e tarefas necessárias para desenvolver um software de alta qualidade. Um processo (metodologia), segundo o IEEE, é uma "sequencia de passos executados com um determinado objetivo"; segundo o CMMI, é "um conjunto de ações inter-relacionadas realizadas para obter um conjunto especificado de produtos, resultados ou serviços" [13].

A escolha do processo de desenvolvimento de software em fabricas de software segue aspectos de conformidade com suas características, ou seja, tamanho da equipe, maturidade em algum processo, agilidade para entrega de produto, necessidade de forte documentação (por conta da alta rotatividade da equipe), estrutura organizacional, todos estes aspectos relacionados e integrados são relevantes para o sucesso dos projetos de software. Portanto, a escolha de um processo é fator primordial para o funcionamento adequado das fabricas de software a fim de buscar melhores resultados em seus projetos.

Dentre os processos, destacam-se os métodos tradicionais e os métodos ágeis, sendo que algumas fábricas de software julgam que o software que produzem pode ser compreendido simplesmente ao ler seu código-fonte (métodos ágeis), outras fábricas, documentam seus produtos de forma intensiva (métodos tradicionais) [20].

Entretanto, centros de desenvolvimento de software de repartições públicas apresentam problemas na adoção de métodos tradicionais ou ágeis, pois possuem características distintas daquelas comumente encontradas nas demais indústrias de software. Dentre estas destacam-se: número reduzido de profissionais, alta rotatividade, processos demorados de licitação etc. Neste sentido, a aplicação dos métodos tradicionais em muitas repartições públicas apresenta como principal problema a necessidade de grandes equipes para atenderem todas as fases e papeis definidos por estes modelos. Já nos processos ágeis que, normalmente são aplicados em empresas com foco comercial (com o proposito a busca por maior competitividade, produtividade e lucro), a aplicação desta metodologia se torna pouco viável em repartições públicas, uma vez que são orientadas a codificação, fica a cargo do desenvolvedor a realização de

códigos auto-documentáveis. Acresce-se que há altos riscos decorrentes da crescente rotatividade das equipes por consequência do mercado aquecido da carreira.

Neste escopo, surge como desafio a escolha de uma metodologia que guie os ciclos de vida de desenvolvimento de sistemas e que seja seguido por seus desenvolvedores, permitindo agilidade e acompanhamento de todo o processo do setor de desenvolvimento em muitas repartições públicas brasileiras.

Visando contornar os problemas apresentados e propor viabilidade de soluções para pequenas equipes, este trabalho propõe um conjunto de técnicas computacionais e gerenciais, adequando a metodologia de processos unificados às características ambientais do setor de desenvolvimento de software em repartições públicas.

II. FUNDAMENTOS

Nas últimas décadas foram criadas várias estratégias de desenvolvimento de software [16], geralmente fundamentadas em metodologias ágeis ou tradicionais. Para [4], a natureza complexa do desenvolvimento de software e a grande variedade de métodos existentes tornam a comparação entre abordagens ágeis e tradicionais difícil e imprecisa. O trabalho em [10] discute essas diferenças, as quais são sumarizadas na Tabela I (extraída de [4]).

TABELA I
Características do paradigma tradicional vs. paradigma ágil segundo [10]

Ponto de vista	Tradicional	Ágil
Medida de sucesso	Conformidade com o plano	Resposta à mudança, código operacional.
Cultura gerencial	Chefia e controle	Liderança / colaboração
Requisitos e arquitetura	Grande e no início	Contínuo / emergente
Garantia de teste e qualidade	Grande, planejado / teste tardio.	Contínuo / concorrente / teste cedo.
Planejamento e cronograma	Detalhado, escopo fixo, tempo e recursos estimados.	Planejamento em dois níveis, data fixa, escopo estimado.

De forma sintética, observa-se que, enquanto o paradigma tradicional tem sido recomendado para projetos de larga escala e alto risco, o paradigma ágil tem se mostrado mais apropriado para projetos de baixo risco e de equipes pequenas [5] [16] [6] [12]. De forma geral, projetos grandes e críticos podem ser prejudicados pela falta de rigor e previsibilidade do paradigma ágil, enquanto que projetos pequenos e de baixo risco podem ter um custo e prazo desnecessariamente elevados pela falta de simplicidade e flexibilidade do paradigma tradicional, que geralmente impõe procedimentos complexos e documentação abrangente. Diante de todos estes conceitos, a adaptação de metodologias se tornou constante para aderência do processo de produção de software. Portanto, fabricas de software sempre buscam customizar os processos de software em conformidade com sua estrutura organizacional.

III. TRABALHOS RELACIONADOS

Vários esforços ocorrem no sentido de adaptar metodologias ágeis e tradicionais a fim de atender as particularidades das organizações. A metodologia adaptada mais conhecida nas indústrias de software é o RUP – Rational Unified Process, que possui as mesmas raízes do Processo Unificado – PU, porém existem várias outras publicações relevantes que utilizam combinação de metodologias ágeis e tradicionais.

Processos de desenvolvimento de software sempre são adequados em conformidade com o ambiente aplicado. Alguns trabalhos relacionados (Tabela II) apresentam adequações de processos ao domínio do desenvolvimento das aplicações, considerando os cenários inerentes a tais domínios, tais como: tamanho e qualificação da equipe de desenvolvimento, estrutura da fábrica de software que hospeda a equipe, experiência da gerência e conjunto de artefatos derivados do processo.

TABELA II
Trabalhos Correlatos

Pesquisa	Descrição	Autor	Ano
P01	Uma Proposta para o Desenvolvimento Ágil de Ambientes Virtuais	Fernando E. R. Mattioli, Edgard A. Lamounier Jr., Alexandre Cardoso	2009
P02	Uma Experiência na Adaptação do RUP em Pequenas Equipes de Desenvolvimento Distribuído	Rodrigo Rocha, Daniel Arcoverde, Rebeka Brito, Bruno Arôxa, Catarina Costa, Fabio Q.B. da Silva, Jones Albuquerque, Silvio Romero de Lemos Meira	2008
P03	Um estudo de caso industrial sobre integração de práticas ágeis no RUP	Nélio Muniz Mendes Albes, William de Souza Carvalho, Alexandre Cardoso, Edgard Afonso Lamounier Junior	2011

A. P01 - Uma Proposta para o Desenvolvimento Ágil de Ambientes Virtuais

Esta proposta aplica-se a ambientes que desenvolvem Sistemas de Realidade Virtual – SRV. Para tal, os autores customizaram a metodologia ágil XP, por ser um processo que agrega características de prototipagem, desenvolvimento iterativo e evolucionário de projetos de software, de forma a atender cinco princípios chaves dos sistemas SRV: 1) A natureza evolucionária dos SRV; 2) A iteratividade da construção e alta fidelidade dos modelos; 3) A necessidade de *feedback* dos clientes; 4) A necessidade de testes de interação e usabilidade; 5) A modularização dos SRV. Para aplicação do processo de desenvolvimento de SRV foram utilizadas seis atividades básicas: Planejamento, Análise, Projeto, Codificação, Testes e Integração. Todas essas atividades são desenvolvidas a cada iteração.

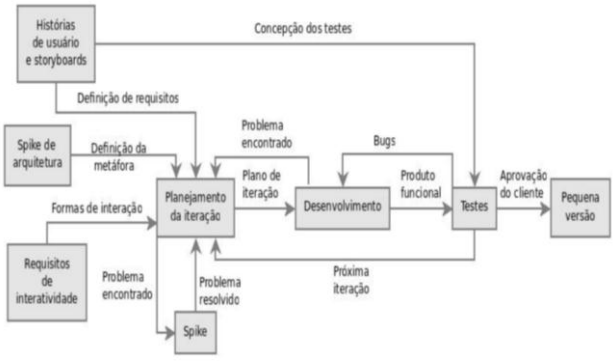

Fig. 1. Ciclo de vida do desenvolvimento ágil de SRV.

O planejamento de uma iteração é feito a partir do *feedback* recebido das iterações anteriores e a cada iteração os modelos do sistema são atualizados. A Figura 1 apresenta o modelo do ciclo de vida do processo ágil de SRV.

Como aspecto relevante, a fase de requisitos de interatividade ocupa posição de destaque dentro do processo de desenvolvimento, uma vez que, na construção de soluções de sistemas de Realidade Virtual é primordial ponderar a usabilidade e a interatividade com o sistema. Entretanto, para a aplicação deste modelo em fabricas de software de repartições públicas para construção do sistema integrado, observa-se como fator limitador o tempo gasto para conceber a fase de requisitos de interatividade desnecessária pela característica do projeto.

B. P02 - Caracterização de um Processo de Software para Projetos de Software Livre

Este trabalho apresenta um método com as mesmas do processo RUP, com a diferença de que elas são realizadas não apenas no escopo do sistema, mas também de cada subsistema, havendo, ainda, um revezamento entre elas ao longo do tempo (Figuras 2). Isso permite que, após a fase de transição (implantação) do(s) primeiro(s) subsistema(s), o conhecimento adquirido e os artefatos produzidos em seu desenvolvimento sejam devidamente explorados nos subsistemas que ainda restam, além de possibilitar melhor condição de avaliação do andamento e das estimativas do projeto como um todo, já que foi adquirida uma amostragem de desenvolvimento do sistema, em cada uma de suas fases.

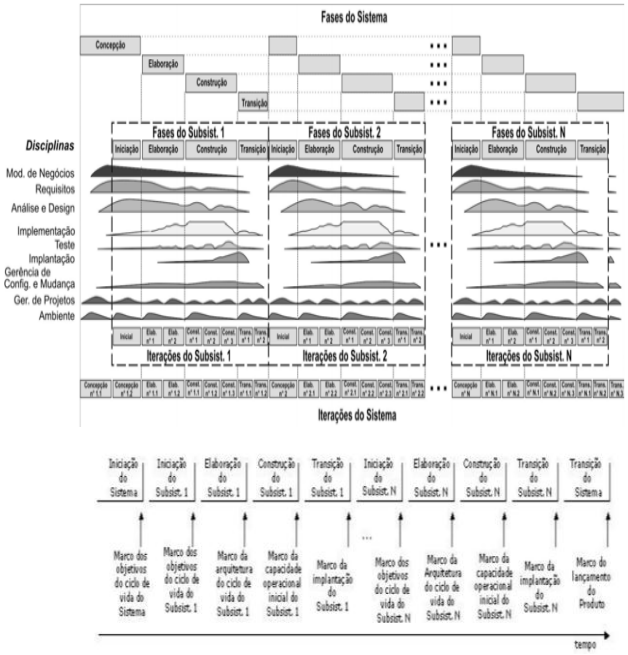

Fig. 2. Ciclo de vida do desenvolvimento RUP para N subsistemas.

Como este processo utiliza todas as fases do RUP em todos os subsistemas, este se torna extremamente inviável na aplicação das características deste estudo de caso, devido a todos os subsistemas ter que repassar por todas as quatro fases que o processo RUP. Isto acarreta uma demanda significativa de recursos humanos para atendimento a todo processo. Para atendimento a este modelo as repartições públicas necessitariam um aumento significativo em seu quadro de pessoal para compor as equipes.

C. P03 - Integração de Princípios de Desenvolvimento Ágil de Software ao RUP – um Estudo Empírico

Esta pesquisa apresentou um estudo sobre a combinação de métodos ágeis (SCRUM) e tradicionais (RUP). Realizou-se um estudo de caso em uma empresa de médio porte, no qual a produtividade de projetos que utilizaram o processo RUP foi comparada com projetos que utilizaram o novo processo híbrido Scrum-RUP.

O processo híbrido Scrum-RUP consiste fundamentalmente da implementação RUP utilizada pela empresa, praticamente mantendo os mesmos papéis e artefatos, porém combinada com práticas ágeis principalmente inspiradas pelo Scrum. As principais características do processo Scrum-RUP, é o fato deste seguir o modelo de ciclo de vida do RUP, com uma etapa de conceito inicial correspondente à fase de iniciação, porém os subprocessos relacionados a requisitos, design, implementação, teste, implantação e gestão são executados por meio de uma Sprint (a empresa utilizou a duração de duas semanas), conforme esboçado na Figura 3. A execução destes subprocessos na Sprint é acompanhada de algumas práticas ágeis que afetam a dinâmica da equipe de desenvolvimento, tais como proximidade física dos colaboradores, maior comunicação, troca de parte da documentação por conhecimento tácito, reuniões diárias e maior colaboração.

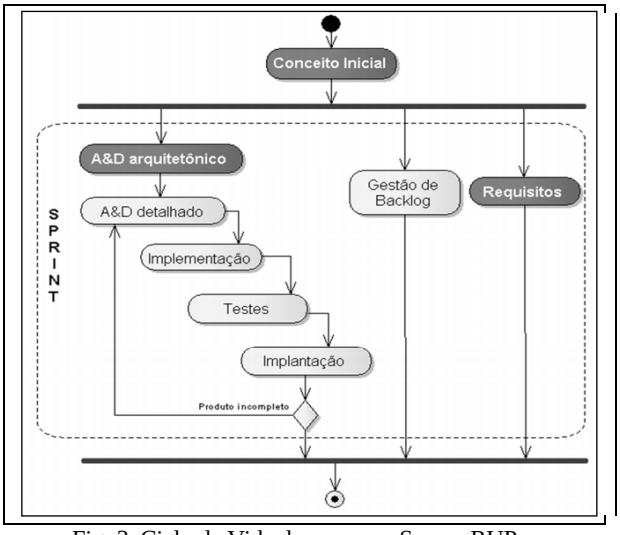

Fig. 3. Ciclo de Vida do processo Scrum-RUP

Para aplicação deste modelo, destacam-se dois aspectos relevantes caso este processo estivesse implantado em ambientes de repartições públicas, tais como: 1) requisitos predominantes por consequência dos clientes on-site; 2) Integração continua: Necessidade por característica do projeto em integrar todos os submódulos para um melhor entendimento do conceito GRP-IFTM.

Para análise comparativa dos três trabalhos relacionados a Tabela III apresenta as principais diferenças dos trabalhas relacionados com o processo proposto.

TABELA III
Tabela comparativa entre os trabalhos

	Ágil SRV (P01)	RUP – N Sistemas (P02)	SCRUM-RUP (P03)	Processo Proposto
Iterações e incrementos	x	x	x	X
Planejamento antecipado	x	X	x	X
Cliente on-site		X		X
Equipes auto-organizáveis	x			
Integração contínua	x			X
Design simplificado		X	x	X
Refatoração	x			
Código próprio da equipe	x	X		X
Estimativa de esforço antecipada			x	X
Reuniões diárias		X	x	X
Arquitetura antecipada		X	x	X
Gestão formal de requisitos		X	x	x

IV. PROPOSTA DESTE TRABALHO

Este trabalho propõe uma adequação do Processo Unificado, visando, primordialmente, customizar o mesmo com sensível redução de número de fases e artefatos, a fim de obter melhor gerenciamento do processo de desenvolvimento de software em fabricas com características de repartições públicas. Espera-se assim contribuir na perspectiva da integração de módulos hierárquicos conceitualmente definidos em projetos corporativos governamentais definidos como GRPs. A Figura 4 apresenta o ciclo de vida como proposta deste trabalho.

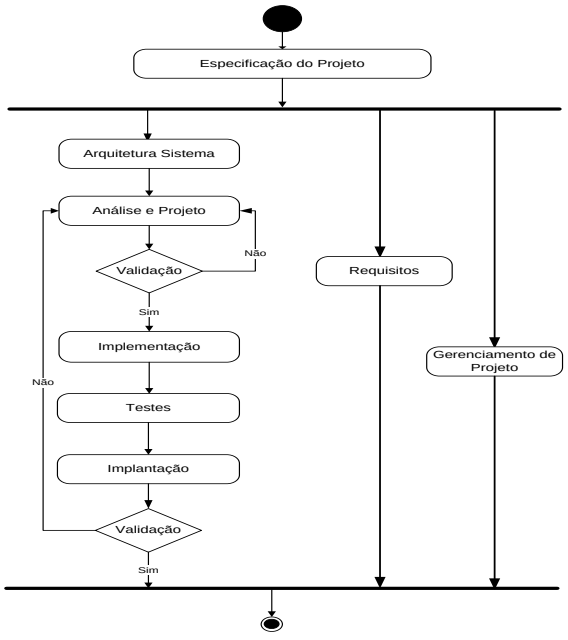

Fig. 4. Modelo de ciclo de vida proposto

O ciclo de vida do processo proposto inicia através da especificação do projeto: nesta fase discute-se o escopo do projeto, seu objetivo e sua finalidade. Nesta fase, também são definidos todos os participantes do submódulo hierárquico proposto (apresentado na seção cenário do projeto) aos quais são divididos entre os papeis conforme demonstrados abaixo:

- **Patrocinador do Projeto**: Para o módulo principal GRP-IFTM, bem como para todos os subprojetos este papel é sem dúvida o de maior importância, pois é a pessoa que acredita no projeto, e tem como principal função motivar e ser o maior responsável pelo sucesso do mesmo.
- **Gerente do Projeto:** O gerente do projeto tem como atividade em destaque a sintonia com todos os outros subprojetos para garantir o alinhamento das integrações a serem realizadas e otimizadas, ajusta as prioridades a serem desenvolvidas, coordena as reuniões com os clientes dos submódulos e define as metas a serem alcançadas pela equipe do projeto.
- **Coordenador de Sistemas:** Este colaborador irá coordenar a equipe de desenvolvedores, aplicando e acompanhando a metodologia proposta, utilizando ferramentas e tecnologias padronizadas e aprovadas.
- **Desenvolvedor:** O papel do desenvolvedor é Implementar, testar e manter todas as funcionalidades em conformidade com os requisitos especificados. Preparar massas de dados a fim de verificar seu correto desempenho, corrigindo possíveis erros. Organizar a documentação pertinente. Realizar as atividades conforme metodologia aplicada, gerando os artefatos necessários para sua documentação.
- ***Stakeholders*:** Este papel representa os grupos de interesse cujas necessidades devem ser atendidas pelo projeto. É um papel que deve ser executado pelas pessoas que possuem conhecimentos da regra de negócio a fim de alimentar os requisitos necessários para construção do módulo.

Ainda na fase de especificação do projeto o principal artefato gerado é o "Termo de Abertura do Projeto – TAP", que defini o escopo, o objetivo e a finalidade do projeto, e este é construído por meio de reuniões (no máximo duas) realizadas com todos os participantes do projeto.

A fase de Análise e Projeto proporciona a iteração com os *stakeholders* envolvidos no projeto. Nesta fase, são realizadas várias reuniões previamente agendadas com proposito de levantar os requisitos funcionais, gerando os artefatos casos de usos e protótipos de telas. Estes artefatos são apresentados aos *stakeholders* para serem aprovados. A iteração desta fase acontece até que todos os casos de usos e protótipos sejam devidamente aprovados, caso contrário este processo permanece no ciclo para ajuste e adequações. A aprovação gera dos casos de uso e dos protótipos gera um importante artefato denominado "Termo de Aceite do usuário".

Após os protótipos e casos de uso aprovados, a fase de implementação tem como principal tarefa a codificação em conformidade com os artefatos gerados na fase anterior (casos de uso e protótipos de telas), este código deverá obedecer a padrões de desenvolvimento previamente construído. Toda implementação é realizada em uma infraestrutura idêntica a infra de produção, ou seja, todos os submódulos e suas funcionalidades são construídos e armazenados em um servidor denominado "servidor teste" e este tem o mesmo comportamento do servidor de produção. Ao término da implementação os desenvolvedores iniciam a fase de testes unitários das funcionalidades requisitadas, e estes são realizados de forma a minimizar problemas de bugs e requisitos incoerentes com o projetado. Além dos testes serem feitos pelos desenvolvedores, estes também são

realizados pelos *stakeholders*, a fim de certificar os requisitos levantados e devidamente aprovados e novamente registrado o aceite por parte do usuário.

A fase de implantação realiza a transferência de toda estrutura do servidor teste para o servidor de produção e neste momento é realizada uma nova iteração com os *stakeholders* para sua aprovação final.

Existem duas fases, nas quais o ciclo de vida do processo proposto é continuo: a fase de "Requisitos", onde iterações entre as equipes de desenvolvimento e os *stakeholders* envolvidos no projeto, acontecem em momentos informais, ou seja, ocorrem através de email, verbal ou outra fonte de comunicação, pois é comum o aparecimento de dúvidas para resolução de problemas que passaram despercebidos na fase do levantamento dos requisitos.

Outra fase continua é o Gerenciamento de Projetos, esta representa o acompanhamento total do projeto. Para isso é utilizado uma ferramenta de gerenciamento de projetos que possibilita o acompanhamento de todas as atividades e de todos os envolvidos nos projetos, bem como o armazenamento de forma organizada de todos os artefatos gerados no processo de desenvolvimento dos submódulos e consequentemente do projeto como um todo.

V. ESTUDO DE CASO

De forma a definir prova de conceito, aplicou-se a metodologia adaptada ao Processo Unificado em três equipes da fabrica de software do Instituto Federal de Educação, Ciência e Tecnologia do Triângulo Mineiro. Será apresentado um contexto global a fim de obter uma melhor compreensão sobre todo o trabalhado aplicado.

A. A Empresa

Os Institutos Federais de Educação, Ciência e Tecnologia foram criados em dezembro de 2008 através da lei 11.892 com o objetivo de atender a educação profissional e tecnológica no contexto social do Brasil. Em consequência as antigas autarquias Centro Federal de Educação Tecnológica de Uberaba – CEFET-Uberaba e Escola Agrotécnica Federal de Uberlândia – EAF-Uberlândia, uniram-se e constituíram a nova instituição, o Instituto Federal de Educação, Ciência e Tecnologia do Triângulo Mineiro– IFTM. O IFTM atualmente conta com Reitoria na cidade de Uberaba e quatro câmpus (Uberaba, Uberlândia, Ituiutaba e Paracatu) e dois câmpus Avançado (Uberlândia e Patrocínio). Atualmente conta com aproximadamente 700 servidores efetivos para atendimento a 3000 alunos presenciais e 5000 alunos a distância.

B. As Equipes

Para atendimento as demandas da área de Tecnologia da Informação e Comunicação, a estrutura organizacional da instituição criou um departamento chamado Diretoria de Tecnologia da Informação e Comunicação, a fim de atender todos os projetos de software do IFTM. Para este estudo de caso, envolveu-se os colaboradores das equipes de sistemas conforme estão organizadas na Tabela IV, ou seja, três equipes com um líder e dois desenvolvedores cada. Estas equipes trabalham em sintonia a fim de sincronizar as ações de desenvolvimento dos projetos de software.

TABELA IV
Identificação das equipes

Identificação da equipe	Qtde de colaboradores
EQ1	Um líder e dois desenvolvedores
EQ2	Um líder e dois desenvolvedores
EQ3	Um líder e dois desenvolvedores

C. Identificação do problema específico do estudo de caso

Os Institutos Federais são organizações novas - criados em dezembro de 2008 - com três anos de funcionamento. Apesar da absorção de antigas autarquias a concepção dos Institutos possuem caraterísticas amplamente diferentes. Consequentemente os modelos administrativos e gerenciais passaram por trágicas mudanças com a implantação desta nova instituição, exigindo soluções emergentes na automatização destes processos.

Além de todo mapeamento realizado no inventário dos softwares existentes no âmbito do IFTM, investigou-se como eram desenvolvidos os sistemas implantados nas antigas autarquias e detectou nenhuma metodologia formalmente registrada, apenas ações individuais, pequenos sistemas eram desenvolvidos para atendimento a particularidades de setores através da metodologia código-produto. Diante disso, não foram encontrados artefatos, ficando como documentações apenas códigos-fontes não estruturados de alguns softwares.

Neste cenário, o IFTM sinalizou vários problemas, tais como: integração de sistemas, protocolos de comunicação entre pessoas e processos; significativo tempo gasto na detecção de solução de problemas nos sistemas dentre outros oriundos da má utilização.

D. O Projeto GRP-IFTM

No cenário apresentado, elaborou-se um projeto chamado GRP-IFTM que tem o objetivo de integrar todos os processos administrativos e gerenciais desenvolvidos no âmbito do IFTM. Este projeto foi dividido em módulos - conforme apresentado na figura 5.

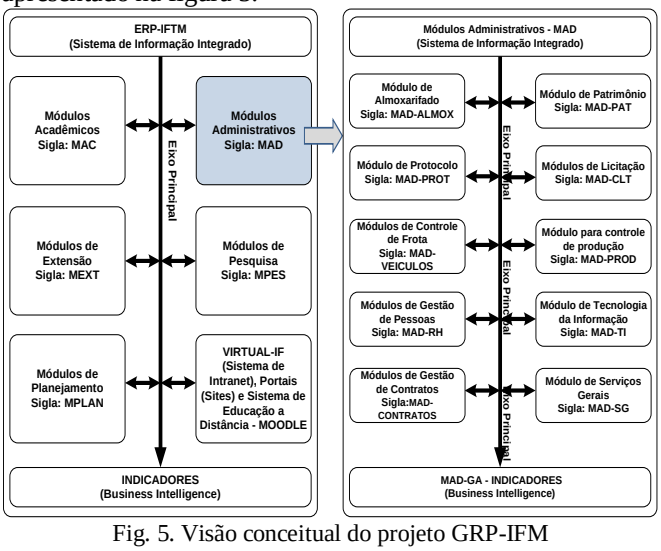

Fig. 5. Visão conceitual do projeto GRP-IFM

Como fontes para este estudo de caso foram separados os submódulos MAD-ALMOX, MAD-RH e MAC-CRA para aplicação da metodologia proposta.

A distribuição dos submódulos para equipes foi organizada conforme Tabela V.

TABELA V.
Distribuição dos submódulos entre as equipes

Equipe	Módulo	Descrição do submódulos
EQ01	MAD-ALMOX	Módulo de Almoxarifado
EQ02	MAC-CRA	Módulo de Controle e Registro Acadêmico
EQ03	MAD-RH	Módulo de Recursos Humanos

VI. APLICAÇÃO DA METODOLOGIA PROPOSTA

A aplicação do processo proposto iniciou por meio de um treinamento com todos os integrantes das três equipes. No treinamento foram apresentadas todas as fases, os papeis e os artefatos que devem ser gerados. Além do treinamento do processo proposto foi implantado e apresentado o ambiente de gerenciamento de projeto denominado Redmine, que é o responsável pelo acompanhamento de todas atividades do projeto, além de permitir o armazenamento dos artefatos produzidos.

Após a aplicação da metodologia, foram levantados alguns riscos que poderiam impactar, consideravelmente, na construção dos módulos do GRP-IFTM:

- Falta de domínio e experiências no uso de outras metodologias por dos membros das equipes;
- Definição concreta dos papeis para realização de todas as atividades do modelo proposto;
- Numero de colaboradores por projetos, menor do que o necessário e alterações destes para outros projetos causando problemas no planejamento inicial das tarefas;
- Possíveis atrasos devido a dependências de disponibilidade dos *stakeholders* envolvidos;
- Controle por parte dos coordenadores em realizar os devidos registros no ambiente;

VII. CONSIDERAÇÕES FINAIS

Este trabalho apresentou como proposta a adequação do processo unificado para desenvolvimento de software. Foi aplicado por meio de estudo de caso no Instituto Federal de Educação, Ciência e Tecnologia do Triângulo Mineiro, com o intuito de oferecer boa aderência ao processo de desenvolvimento, refletindo em economia de tempo na geração de artefatos. Os resultados obtidos foram satisfatórios, pois a aplicação do processo no desenvolvimento do GRP-IFTM proporcionou uma melhor gerência do projeto e sua organização, ao permitir definição de prazos mais coerentes, de cronogramas adequados, de uma maior integração entre as equipes, de definição dos papeis e na certeza de uma maior facilidade na manutenção do sistema proporcionado por toda documentação do projeto.

VIII. TRABALHOS FUTUROS

Este trabalho encontra-se em fase de adequações para validação de tais técnicas, portanto pontos importantes devem ser abordados em trabalhos futuros, como por exemplo, o levantamento de dados qualitativos de produtividade para certificar se a adoção desta proposta conseguiu aprimorar a capacidade de produção de software de seus colaboradores. Outro aspecto relevante é a aplicação do *framework* MPS.BR (Melhoria de Processos do Software Brasileiro) para melhorar a capacidade de desenvolvimento do GRP-IFTM.

REFERÊNCIAS BIBLIOGRÁFICAS

[1] S. W. AMBLER, *Agile Software Development at Scale, IFIP European Conference on Software Engineering Techniques*. Poznan: Springer. 2007.

[2] K. BECK, *Extreme Programming Explained: Embrace Chage*. 2. ed. Reading: Addison-Wesley, 2004.

[3] K. BECK et al. (2001), *Manifesto for Agile Software Development*. Acedido em 14 de Abril de 2012, em: <http://www.agilemanifesto.org>.

[4] N. ALVES, W. CARVALHO, E. A. LAMOUNIER JR, E. *Um estudo de caso industrial sobre integração de práticas ágeis no RUP*. 2011

[5] B. BOEHM, R. TURNER, *Observations on Balancing Discipline and Agility*. Agile Development Conference. Salt Lake City, Utah, USA: IEEE Computer Society. 2003.

[6] M. COHN, *User Stories Applied: For Agile Software Development. Reading*: Addison-Wesley, 2004.

[7] R. ROCHA et atl., *Uma Experiência na Adaptação do RUP em Pequenas Equipes de Desenvolvimento Distribuído*.

[8] F. MATTIOLI, et al.: *Uma Proposta para o Desenvolvimento Ágil de Ambientes Virtuais*. 2010

[9] P. KRUCHTEN, *Introdução ao RUP – Rational Unified Process*. Editora Ciência Moderna Ltda. 2. ed., Rio de Janeiro, 2003.

[10] D. LEFFINGWELL., *Scaling Software Agility*. Reading: Addison Wesley, 2006

[11] F. COALIER, M. AZUMA, "Introduction to Software Engineering Standards", *IEEE Computer Society*, Estados Unidos, 1998.

[12] R. NORD, J. TOMAYKO, S*oftware Architecture-Centric Methods and Agile Development*. IEEE Software, 2006.

[13] W. P. F. PÁDUA, E*ngenharia de Software. Fundamentos, Métodos e Padrões*. LTC – Livros Técnicos e Científicos, 3º Ed. – Rio de Janeiro, 2009.

[14] R. PRESSMAN, *Engenharia de Software*. McGraw-Hill, 7ª, ed., São Paulo, 2011

[15] S. L. PFLEEGER, *Engenharia de software :teoria e prática*, Prentice Hall, 2ª ed. São Paulo, 2004.

[16] R. RAMSIN, R. F. PAIGE, *Process-Centered Review of Object Oriented Software Development Methodologies*. ACM Computing Surveys, v. 40, n. 1, 2008.

[17] S. W. AMBLER, *Modelagem Ágil – Práticas eficazes para a programação extrema e o processo unificado*, Bookman, 1ª ed. Porto Alegre-RS, 2004.

[18] K. SCHWABER, J. SUTHERLAN (2010), *Scrum Guide,* Acedido em 24 de Abril de 2012, em: Scrum.org.

[19] I. SOMMERVILLE, *Engenharia de Software*, Pearson, 9ª ed., São Paulo, 2011.

[20] S.R. SCHACH, *Engenharia de software : Os Paradigmas Clássicos e Orientado a Objetos*, Mc Graw Hill, 7ª ed., São Paulo, 2009.