O Processo Unificado da Rational conhecido como RUP (Rational Unified Process), é um processo de engenharia de software criado para apoiar o desenvolvimento orientado a objetos, fornecendo uma forma sistemática para se obter vantagens no uso da UML. Foi criado pela Rational Software Corporation e adquirido em fevereiro de 2003 pela IBM.

O principal objetivo do RUP é atender as necessidades dos usuários garantindo uma produção de software de alta qualidade que cumpra um cronograma e um orçamento previsíveis. Assim, o RUP mostra como o sistema será construído na fase de implementação, gerando o modelo do projeto e, opcionalmente, o modelo de análise que é utilizado para garantir a robustez. O RUP define perfeitamente quem é responsável pelo que, como as coisas deverão ser feitas e quando devem ser realizadas, descrevendo todas as metas de desenvolvimento especificamente para que sejam alcançadas.

O RUP organiza o desenvolvimento de software em quatro fases, onde são tratadas questões sobre planejamento, levantamento de requisitos, análise, implementação, teste e implantação do software. Cada fase tem um papel fundamental para que o objetivo seja cumprido, distribuídos entre vários profissionais como o Analista de sistema, Projetista, Projetista de testes, entre outros.

Melhores práticas:

1. **Desenvolver o software iterativamente**: planejar os incrementos de software com base nas prioridades do cliente e desenvolver e entregar o mais cedo possível às características de sistema de maior prioridade no processo de desenvolvimento.
2. **Gerenciar Requisitos**: documentar explicitamente os requisitos do cliente e manter acompanhamento das mudanças desses requisitos. Analisar o impacto das mudanças no sistema antes de aceitá-las.
3. **Usar arquiteturas baseadas em componentes**: Estruturar a arquitetura do sistema com componentes, reduzindo a quantidade de software a ser desenvolvido e, consequentemente, reduzir custos e riscos.
4. **Modelar software visualmente**: usar modelos gráficos de UML para apresentar as visões estática e dinâmica do software.
5. **Verificar a qualidade do software**: garantir que o software atenda aos padrões de qualidade da organização.
6. **Controlar as mudanças do software**: gerenciar as mudanças do software, usando um sistema de gerenciamento de mudanças, procedimentos e ferramentas de gerenciamento de configuração.

Fases do RUP:

1. **Concepção**: o objetivo desta fase é estabelecer um business case para o sistema. Devem ser identificadas todas as entidades externas (pessoas e sistemas) que irão interagir com o sistema em desenvolvimento e definir essas interações. Essas informações são utilizadas para avaliar a contribuição do novo sistema para o negócio.
2. **Elaboração**: os objetivos desta fase são desenvolver um entendimento do domínio do problema, estabelecer um framework de arquitetura para o sistema, desenvolver o plano de projeto e identificar seus principais riscos. Ao final desta fase deve-se ter um

modelo de requisitos para o sistema (os casos de uso da UML são especificados), uma descrição de arquitetura e um plano de desenvolvimento do software.

3. **Construção**: está fase está essencialmente relacionada ao projeto, programação e teste do sistema. As partes do sistema são desenvolvidas paralelamente e integradas durante esta fase. Ao final deve-se ter um sistema de software em funcionamento e a documentação associada pronta para ser liberada para os usuários.
4. **Transição**: nesta fase, faz-se a transferência do sistema da comunidade de desenvolvimento para a comunidade de usuários, com a entrada do sistema em funcionamento no ambiente real. Esta é uma atividade ignorada na maioria dos modelos de processo de software, pois é onerosa e às vezes problemática. Ao final desta fase, deve-se ter um sistema de software documentado, funcionando corretamente em seu ambiente operacional.

Workflows:

- **Modelagem de Negócios**: Os processos de negócio são modelados usando casos de uso de negócios.
- **Requisitos**: Os agentes que interagem com o sistema são identificados e os casos de uso são desenvolvidos para modelar os requisitos do sistema.
- **Análise e Projeto**: Um modelo de projeto é criado e documentado usando modelos de arquitetura, modelos de componente, modelos de objetos e modelos de sequencia.
- **Implementação**: Os componentes de sistema são implementados e estruturados em subsistemas de implementação. A geração automática de código com base nos modelos de projeto ajuda a acelerar esse processo.
- **Teste**: O teste é um processo iterativo realizado em conjunto com a implementação. O teste de sistema segue o término da implementação.
- **Implantação**: Uma versão do produto é criada, distribuída aos usuários e instalada no local de trabalho.
- **Gerenciamento de Configuração e Mudança**: Este workflow de apoio gerencia mudanças no sistema.
- **Gerenciamento de Projetos**: Este workflow de apoio gerencia o desenvolvimento do sistema.
- **Ambiente**: Este workflow está relacionado à disponibilização de ferramentas apropriadas de software para a equipe de desenvolvimento.

Workflows: Cada workflow é descrito em detalhe, apresentando passo a passo as tarefas, subprodutos a serem gerados e papéis de profissionais envolvidos. Cada tarefa, subproduto e papel é descrito em detalhe. Este modelo pode ser seguido à risca ou adaptado conforme sua necessidade específica.

Tarefas: Cada tarefa é descrita em detalhe, incluindo que papel é responsável por ela, a qual workflow ela pertence e quais são os subprodutos de entrada e saída.

Modelo de equipe: Os diversos papéis necessários no projeto são descritos em detalhe. Assim como no MSF, cada papel pode ser representado por mais de uma pessoa, ou uma pessoa pode ter mais de um papel, dependendo da carga de trabalho necessário. Porém o RUP aborda

os papéis em um maior nível de detalhe. Ao todo são mais de 30. Exemplos de papéis são: analista de sistemas, analista de negócio, etc.

Modelos de documentos: O RUP apresenta modelos e exemplos para os diversos documentos (artifacts) gerados ao longo do projeto. Estes documentos são descritos em detalhe, assim como as tarefas que os geram e as que os utilizam. Para os usuários das ferramentas Rational, existe um recurso adicional e-coach, que orienta o usuário a usar ferramentas como o Requisite Pro passo a passo. Os documentos são totalmente compatíveis com a UML, o que reforça a questão de padronização.

Métodos concorrentes no campo da engenharia de software incluem o "Cleanroom" (considerado pesado) e os Métodos Ágeis (leves) como a Programação Extrema (XP-Extreme Programming), Scrum, FDD e outros.

Embora o RUP não seja um processo adequado a todos os tipos de desenvolvimento de software, ele representa uma nova geração de processos genéricos. A mais importante inovação é a separação de fases e workflows, e o reconhecimento de que a implantação de software no ambiente do usuário é parte do processo. As fases são dinâmicas e tem objetivos. Os workflows são estáticos e constituem atividades técnicas que não estão associadas a uma única fase, mas podem ser utilizados ao longo do desenvolvimento para atingir os objetivos de cada fase.

Bibliografia

- http://pt.wikipedia.org/wiki/IBM_Rational_Unified_Process
- http://www.wthreex.com/rup/process/ovu_proc.htm
- http://www.wthreex.com/rup/process/workflow/environm/co_morop.htm
- http://www.tiespecialistas.com.br/2011/02/rup-primeiros-passos/
- http://www.linhadecodigo.com.br/artigo/79/conheca-o-rational-unified-process-rup.aspx
- http://www.infoescola.com/engenharia-de-software/rup/