Crystal Clear Methodologies

Thiago Sinésio, Briner Nunes Homem, Carlos Alexandre Correia, Daniel Augusto da Silva

Centro Universitário UNA
Caixa Postal 30140-071 – Belo Horizonte – MG – Brasil

Curso de Pós Graduação em Engenharia de Software Centrada em Métodos Ágeis

thiagosinesio@gmail.com, *briner.nunes@cdlbh.com.br, ccorreiax@gmail.com, danielccunibh@gmail.com*

Abstract

Crystal family is a group of methodologies created by Alistair Cockburn. The methodology has approach to people management. As Crystal Family is very sensitive to human factors, it is not purposely completely defined, been necessary to adapt it to each project. To chose which method use, we should consider the number of people and the criticality of the project. Crystal Clear is a Crystal's family member, and was done to projects with two to eight people, sitting in the same room or near offices, so all people can easily communicate with each other.

Resumo

A Família Crystal é um conjunto de metodologias criada por Alistair Cockburn. A metodologia possui uma abordagem voltada à gestão de pessoas. Como Crystal Family é muito sensível a fatores humanos, ela não é, propositalmente, completamente definida, devendo se adaptar a cada projeto. Para a escolha de qual metodologia usar deve-se considerar o número de pessoas e a criticidade do projeto. Crystal Clear é membro da família Crystal, e é voltada para projetos com duas a oito pessoas, sentadas na mesma sala ou escritórios próximos, de forma que todas as pessoas possam se comunicar facilmente.

1. Introdução a Metodologias Ágeis

A indústria de desenvolvimento de software é uma das mais importantes dos tempos atuais. Além de empregar milhares de trabalhadores em todo o mundo, cria e participa de produtos essenciais para nossa sobrevivência Günal (2012).

Devido ao constante desenvolvimento dos sistemas de informação, a crescente demanda de seus serviços, bem como as diferentes formas que o mundo dos negócios assumiram ao longo dos anos, diferentes métodos e modelos de desenvolvimento de software foram inventados e utilizados nas últimas cinco décadas Günal (2012).

Até o ano de 2001, a maioria dos modelos de processo de desenvolvimento de software eram baseados no modelo "Cascata", onde há uma grande fase de planificação que da suporte a todo o restante do projeto. Através da utilização desse modelo, qualquer alteração de requisito exige regressão à fase de planificação, onde pode causar grande impacto em todo o projeto, esse fator faz com que as empresas tenham grande dificuldade em aceitar mudanças nos requisitos nas fases posteriores ao planejamento Tomás (2009).

Alguns métodos tradicionais são muito orientados a documentação, limitando os desenvolvedores a seguir e aplicar determinados processos. Como reação a esses problemas trazidos pelos métodos tradicionais, começou a surgir na indústria de software ao longo das últimas duas décadas a compreensão da importância do fator humano, importância da colaboração e comunicação entre a equipe de desenvolvimento e os clientes e a capacidade de responder as mudanças Günal (2012); Agile Manifesto (2001).

Filho (2011) relata que o modelo de desenvolvimento em Cascata está correlacionado com falhas no projeto e redução da produtividade.

Com o grande crescimento das demandas por software se faz necessário criar e entregar mais rapidamente os produtos de software com mais qualidade e que melhor reflitam as reais necessidades dos clientes Libardi e Barbosa (2010).

Libardi e Barbosa (2010), argumentam que o desenvolvimento de software precisa ser reconhecido como um processo complexo. A grande mudança no pensamento tradicional do processo de desenvolvimento de software é reconhecer que os softwares não são construídos com a mesma equipe, da mesma forma, sob as mesmas circunstâncias e ter a consciência que durante todo o processo pode haver imprevistos e mudanças a equipe precisa ter ações para trabalhar nessas situações.

As ideias que guiam o desenvolvimento ágil tiveram início antes da década de 90, porém só em 2001 os princípios desta metodologia foram publicados sob a forma de manifesto. O manifesto denominado "Manifesto for Agile Software *Developer*" define 12 princípios, a partir dos quais foram criados os "Modelos Ágeis de Processo", dentre eles a *Crystal Clear Methodologies* Tomás (2009).

Highsmith (2002) define agilidade como a habilidade de criar e responder a mudanças, de modo a lucrar em um ambiente turbulento de negócios.

2. Introdução a *Crystal Family*

A família *Crystal* foi criada por Alistair Cockburn e possui uma abordagem voltada à gestão de pessoas, propositalmente pouco definida e muito sensível a fatores humanos, focados nas habilidades e talentos das pessoas. Seus princípios são adaptados para cada projeto de acordo com sua complexidade, adotando um conjunto de políticas e convenções para cada situação.

A *Crystal Methodologies*, é uma família de metodologias com um código genético em comum, que podem ser adaptados de acordo com o projeto ou o número de integrantes da equipe. Sendo assim, é importante ressaltar que não há "uma" metodologia *Crystal*. Existem diferentes tipos de metodologias *Crystal* para diferentes tipos de projeto.

Apesar de ser pouco definida, a família *Crystal* possui valores comuns, como: entregas frequentes; equipe com especialidades distintas e com boa proximidade e integração; não limitação de quaisquer práticas de desenvolvimento; visa comunicação eficaz e bom fluxo de informações; os projetos possuem ciclo de desenvolvimento incremental e com liberação de versões variando de um a quatro meses. Após cada iteração são feitas reflexões sobre possíveis melhorias. Cada organização avalia o projeto por duas visões: número de pessoas e consequência dos erros.

Como a metodologia não é detalhadamente definida e varia de acordo com o projeto, utilizam-se cores para indicar a complexidade ou "dureza" da metodologia, sendo que, quanto mais escura, mais complexa ela é. Sendo assim, a escolha da cor é baseada no número de pessoas que necessitam ser coordenadas e na criticidade do projeto, ou seja, o grau do dano que pode ser causado pelo seu mau funcionamento. O nome "*Crystal*" caracteriza os projetos através de duas dimensões: tamanho e criticidade, que correspondem a cor e dureza dos minerais.

Crystal Clear é considerada uma metodologia leve, sendo assim para equipes de 2 a 8 pessoas, podendo chegar até 12 em casos especiais utiliza-se "*Clear*"; "*Yellow*", para equipes por volta de 10 a 20 membros; "*Orange*" e "*Orange Web*" são apropriados para times de 20 a 75 participantes; "*Red*" para equipes de 75 a 100 membros; e "*Maroon*" de 200 para cima.

É preciso analisar três fatores para o entendimento de como proceder na adaptação e classificação por cor: a coordenação e comunicação (representada pelo número de pessoas); criticidade do sistema e prioridade do projeto Cockburn (2004).

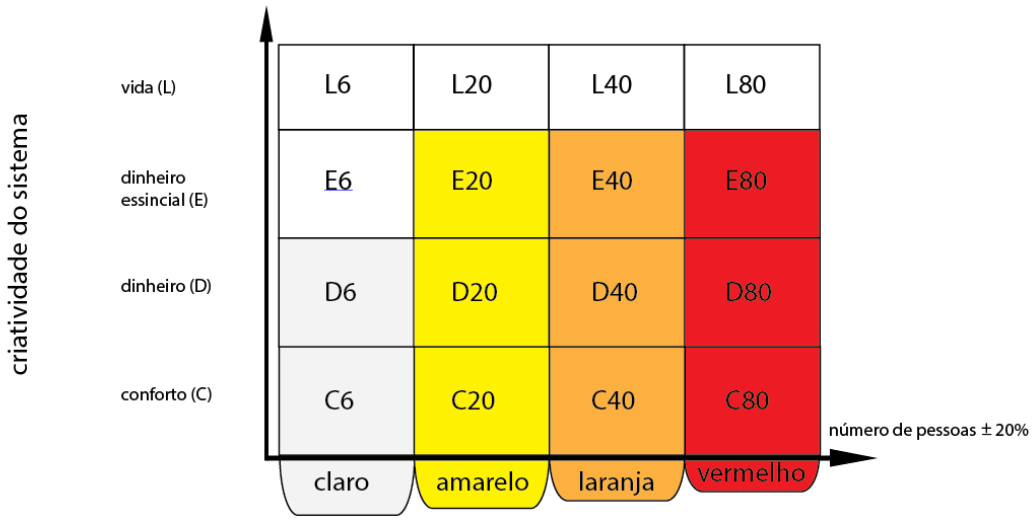

Figura 1. Demostra a classificação por cores das Metodologias *Crystal*, sendo que quanto mais escura é a cor mais complexo e crítico será o projeto.

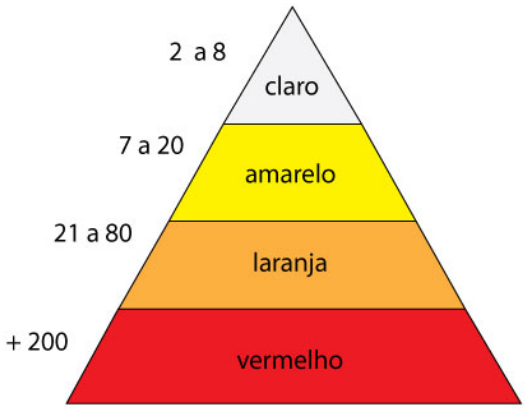

Figura 2. Ilustra a relação da escolha da cor com a quantidade de pessoas no projeto.

3. Código genético da Família *Crystal*

O código genético da Família *Crystal* baseia-se em:

- Modelo de jogo econômico-cooperativo;
- Prioridades selecionadas;
- Propriedades selecionadas;
- Princípios selecionados;
- Técnicas selecionadas;

3.1.Modelo de jogo econômico-cooperativo

O modelo de jogo econômico-cooperativo diz que o desenvolvimento de *software* é uma série de "jogos" cujos movimentos consistem basicamente em invenção e comunicação.

Cada jogo tem dois objetivos conflitantes: entrega de *software* funcional e preparação para a próxima etapa do jogo. O jogo nunca se repete, de modo que cada projeto exige estratégias ligeiramente diferentes de todos os jogos anteriores.

O modelo de jogo econômico-cooperativo leva as pessoas a pensarem sobre seu trabalho em um projeto de uma forma muito específica, focada e eficaz Cockburn (2004).

3.2.Prioridades selecionadas

- Segurança no resultado do projeto (entrega do *software*);
- Eficiência no desenvolvimento;
- Habitabilidade das convenções (o time precisa aceitar o processo).

A principal prioridade é a primeira. As duas últimas são conflitantes, um processo eficiente pode ser intolerável pelas pessoas Cockburn (2004).

3.3.Propriedades selecionadas

A Família *Crystal* enfoca sete propriedades Entrega Frequente (*Frequent Delivery*); Comunicação Cara a Cara (*Close Communication*); Melhoria Reflexiva (*Reflective Improvement*); Segurança Pessoal (*Personal Safety*); Foco (*Focus*); Fácil acesso a especialistas (*Easy Access to Expert Users*); Ambiente técnico com testes automatizados, Gerenciamento de Configuração e Integração Continua (*technical environment with automated testing, configuration management and frequent integration*) Cockburn (2004).

Os três primeiros devem estar presentes em todos projetos.

3.3.1. Propriedade 1: Entrega frequente (*Frequent Delivery*)

A propriedade mais importante de qualquer projeto, grande ou pequeno, ágil ou não, é o de garantir entrega funcional e código testado a cada poucos meses.

As vantagens são inúmeras:

- Os patrocinadores recebem *feedback* sobre a taxa de progresso da equipe;
- Os usuários tem a chance de ver seus produtos em funcionamento ainda na fase de desenvolvimento e assim conseguem saber se o que está sendo feito está de acordo com suas necessidades;
- Desenvolvedores mantem seu foco;
- A equipe se mantém motivada através de realizações.

3.3.2. Propriedade 2: Melhoria Reflexiva (*Reflective Improvement)*

Uma das principais realizações dessa propriedade é que um projeto pode reverter suas falhas catastrófica em sucesso. A equipe se reúne para discutir o que pode funcionar

melhor e fazer essas mudanças na próxima iteração. Em outras palavras, refletir e melhorar.

A equipe não tem que gastar muito tempo para a realização dessa propriedade. Uma hora em algumas semanas ou meses é suficiente. O fato de tomar um tempo diário para pensar no que poderia melhorar já é eficaz Cockburn (2004).

3.3.3. Propriedade 3: Comunicação Cara a Cara (*Close Communication*)

A comunicação cara a cara ou comunicação osmótica é a informação que surge naturalmente entre os membros do time, que conseguem captar as partes relevantes de todo o processo, como se estivessem pegando a informação por osmose.

Isso geralmente acontece pelo simples fato de toda a equipe estar sentada na mesma sala. Assim, quando uma pessoa faz uma pergunta, outros na sala podem contribuir para a discussão ou continuar com o seu trabalho e simplesmente ouvir a resposta.

A comunicação cara a cara é a maneira mais barata e rápida de trocar informações Cockburn (2004).

3.3.4. Propriedade 4: Segurança Pessoal (*Personal Safety*)

A Segurança Pessoal se refere a possibilidade de dizer quando algo está incomodando, sem medo de represálias. Exemplos disso seria dizer ao gerente que o cronograma não está bem feito, ou a um colega que seu projeto precisa ser melhorado.

Segurança Pessoal é importante para a equipe descobrir e trabalhar suas fraquezas. Sem ela, as pessoas não vão falar e as fraquezas vão continuar prejudicando a equipe e o projeto Cockburn (2004).

3.3.5. Propriedade 5: Foco (Focus)

O Foco é fundamental para que os membros da equipe saibam em que trabalhar, qual a prioridade, e ter a tranquilidade para focar naquela tarefa específica. Geralmente o direcionamento do trabalho é dado pelo Patrocinador Executivo (Executive Sponsor).

Essa propriedade destaca que a equipe precisa de tranquilidade para trabalhar na tarefa que lhes foi passada. E essa tranquilidade vem de um ambiente onde as pessoas não são tiradas de sua tarefa para atuar em outras coisas incompatíveis Cockburn (2004).

3.3.6. Propriedade 6: Fácil acesso a especialistas (*Easy Access to Expert Users*)

O fácil acesso a especialistas permite que a equipe possa realizar testes e entregas frequentes; permite rápido *feedback* sobre a qualidade do produto que está sendo desenvolvido e facilita a tomada de decisões Cockburn (2004).

3.3.7. Propriedade 7: Ambiente técnico com testes automatizados, Gerenciamento de Configuração e Integração Continua (*technical environment with automated testing, configuration management and frequent integration*)

3.3.8. Reflexão sobre as propriedades

As sete propriedades não garantem que os projetos estejam na zona de segurança o tempo todo. A Metodologia *Crystal Clear* é construída em torno de propriedades críticas em vez da especificação dos procedimentos.

Uma equipe de *Crystal* trabalha para definir as sete propriedades, usando suas próprias convenções, técnicas e normas para se adequar à sua situação. As convenções podem variar de acordo com cada projeto Cockburn (2004).

3.4.Princípios selecionados

3.4.1. Diferentes projetos precisam de metodologias distintas

Esse princípio prega que projetos diferentes não podem usar sempre a mesma metodologia padronizada, sendo que dois fatores são cruciais para a escolha de qual metodologia usar: o número de pessoas que precisaram ser coordenadas e o grau do dano que pode ser causado pelo mau funcionamento do sistema Cockburn (2004).

3.4.2. Equipes maiores precisam de mais modos de comunicação

Como o tamanho da equipe é o principal fator de escolha de qual metodologia usar, o cuidado com a boa comunicação é essencial. Para *Crystal Clear* prega-se que a equipe deve estar na mesma sala de modo que possua uma boa comunicação osmótica Cockburn (2004).

3.4.3. Quanto mais crítico o projeto, maior deve ser a cerimônia

Pode não ser possível eliminar todos os produtos intermediários relacionados ao trabalho e documentações, como: requisitos, documentos de projeto e planos de projeto porém, eles podem ser reduzidos na medida em que a equipe possui vias de comunicação próximas e informais e o software é testado e entregue frequentemente, sendo que, quanto maior a troca de experiências entre a equipe, menor será a necessidade de documentação. Quanto maior e crítico é o projeto, maior deve ser o cuidado nas decisões, as etapas devem ser mais planejadas e detalhadas.

Crystal Clear não trata nativamente elementos críticos a nível de dano, o time deve ter validações adicionais de acordo com a necessidade Cockburn (2004).

3.4.4. Excesso de metodologia é custo

Deve-se evitar burocracia desnecessária e a escolha da metodologia sempre deve ser a mais leve possível, esse fator leva a menos custos com tempo, ferramentas, motivação e coordenação de pessoas Cockburn (2004).

3.4.5. Formalidade, processo e documentação não substituem habilidade, disciplina e entendimento

Prática vs. Teoria. Processo não é disciplina: Disciplina é trabalhar de forma consistente; Processo é seguir passos já definidos; *Crystal* não incentiva o copiar/colar (e sim o ajustar). Formalidade não é habilidade: só porque há um processo formal o indivíduo não vai ser bom automaticamente; a habilidade (talento individual) é importante. Documentação não necessariamente é entendimento.

O que importa é o entendimento, documentação é importante como meio de comunicação do presente com o futuro Cockburn (2004).

3.4.6. Interatividade e comunicação cara a cara

A comunicação cara a cara é a maneira mais fácil e rápida de trocar informação. Quanto mais pessoas mais necessário será diferentes canais de comunicação.

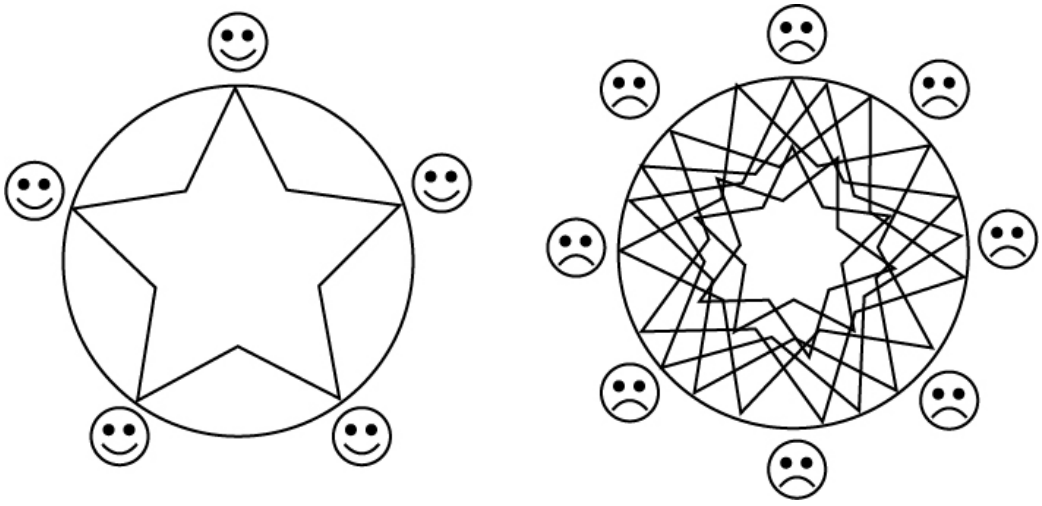

Figura 3. Ilustra a eficiência da comunicação de acordo com o tamanho da equipe.

Crystal Clear favorece o progresso do projeto e reduz o custo, mantendo a comunicação osmótica.

3.4.7. Mais *feedback* reduz a necessidade de entregas intermediárias

As entregas intermediárias aqui se referem a documentos de análise, documentos de requisitos, planos de testes, lista de riscos. Considera-se *feedback* entrega de *software* funcionando.

3.4.8. Desenvolvimento simultâneo e em série reduz custos e promove aumento de velocidade e flexibilidade

Este princípio prega que o desenvolvimento simultâneo pode acelerar o desenvolvimento, mas pode exigir um custo salarial maior, quando comparado com o desenvolvimento em série em que cada tarefa é executada até o final, antes de começar a próxima.

O problema do desenvolvimento em série é que qualquer erro pode provocar um efeito em cascata de retrabalho, que é caro. Sendo assim a Família *Crystal* foi construída para trabalhar com o desenvolvimento simultâneo, que permite descobrir erros rapidamente, reduzindo custo, porém esse tipo de desenvolvimento exige melhor comunicação entre as pessoas Cockburn (2004).

3.4.9. A eficiência é dispensável em atividades que não possuem gargalo

A metodologia destaca que os gargalos devem ser encontrados e que não adianta a equipe tentar otimizar o que não é um gargalo, pois, isso não vai melhorar o projeto como um todo Cockburn (2004).

3.5. Estratégias e Técnicas

Crystal Clear não requer qualquer estratégia ou técnica, porém é aconselhável utilizar algumas.

As estratégias e técnicas listadas abaixo são recomendadas para *Crystal Clear*, principalmente nos primeiros meses do projeto até chegar nas entregas frequentes Cockburn (2004).

3.5.1. As estratégias

a) **Exploratório 360º (*Exploratory 360°)*:** analisar o projeto em todas as direções, checando, por exemplo: viabilidade, valor do negócio, tecnologia. Leva alguns dias até uma ou duas semanas, com base na análise decide-se, por exemplo, se faz sentido usar determinada tecnologia.

b) **Vença Cedo (*Early Victory*):** é uma estratégia de gerenciamento de projetos, que trabalha com a entrega de algo de valor logo no início do projeto para ganhar autor-confiança e a confiança do cliente na equipe.

c) **Esqueleto Andante (*Walking Skeleton*):** implementar uma pequena funcionalidade no sistema do início ao fim, validando a arquitetura e obtendo o *feedback* mais rápido do cliente, que possibilita encontrar problemas mais cedo.

d) **Rearquitetura Incremental (*Incremental Rearchitecture*):** a partir do *Walking Skeleton*, evolução da tecnologia e mudanças nos requisitos a arquitetura do sistema tem que evoluir ao longo do tempo.

e) **Radiadores de Informação (*Information Radiators*):** se refere a deixar as informações do projeto em um lugar visível, possibilitando a equipe sanar todas as dúvidas. Isso significa mais comunicação com menos interrupções.

3.5.2. As Técnicas

a) **Metodologia de Formação (*Methodology Shaping*):** coleta de informações sobre as experiências anteriores, utilizando-as nas convenções iniciais.

b) **Oficina de Reflexão (*Reflection Workshop*):** a equipe deve fazer uma pausa de uma hora periodicamente, após cada entrega, para refletir sobre o trabalho realizado e realizar pequenas alterações no que for necessário para a próxima entrega.

c) **Planejamento Relâmpago (*Blitz Planning*):** semelhante ao "*planning game*" do *XP*, porém deve-se fazer uma lista de dependência entre as tarefas.

d) ***Delphi Estimation***: uma maneira de chegar a uma estimativa inicial para o total do projeto.

e) **Reuniões Diárias (*Daily Stand-ups Meetings*)**: maneira rápida de difundir a informação pela equipe, ela não deve ser usada para discutir problemas e sim para identifica-los.

f) **Design Ágil de Interfaces (*Agile Interaction Design*):** uma versão rápida do *design* centrado no usuário.

g) **Miniatura do Processo (*Process Miniature*):** técnica de aprendizagem rápida sobre o processo que será usado.

h) **Programação lado-a-lado (*Side-by-Side Programming*):** uma alternativa menos intensa do "*Pair Programming*".

i) ***Burn Charts***: maneira de dar visibilidade ao andamento do projeto, através de relatórios.

4. Crystal Clear

Crystal Clear é uma parte da Família *Crystal*, que pode ser aplicada quando a equipe possui de duas a oito pessoas, sentada na mesma sala ou escritórios ao lado, de forma que todas possam se ouvir em qualquer tipo de comunicação sobre o projeto, favorecendo a Comunicação Osmótica, melhor descrita abaixo.

Como a metodologia *Crystal Clear* não é completamente especificada (propositalmente), o primeiro passo para sua adoção é descobrir os pontos fortes e fracos da organização, para adaptar a *Crystal Clear* em torno deles e aproveitar os pontos fortes e cobrir as fraquezas.

A *Crystal Clear* não é feita para empresas que querem coisas padronizadas, ela se destina às organizações que querem construir sua própria metodologia, de maneira eficaz que possibilite entregar *software* rapidamente Cockburn (2004).

5. Processo

A *Crystal Clear* utiliza processos cíclicos de diferentes tamanhos: desenvolvimento, iteração, prazo de entrega e o projeto como um todo.

O ciclo do projeto tem três partes:

a) Mapeamento de atividades

Esta atividade leva de alguns dias a algumas semanas e é constituída por quatro passos:

1) Construir o núcleo da equipe;

   A equipe possui os seguintes papéis:

   - *Executive Sponsor* (patrocinador);
   - *Lead Designer*: deve ser um especialista, com conhecimento de todo o projeto, dentre suas atribuições ele deve: conversar com o patrocinador, treinar membros da equipe menos experientes.
   - *Ambassador User*: é um desenvolvedor especialista que a equipe pode consultar quando precisar.

   A *Crystal Clear* específica somente os três papéis acima, os outros papéis são de acordo com a necessidade do projeto e podem ser: *Designer-Programmer, Business Expert, Coordinator* (deve ser alguém com boa comunicação, geralmente o Lead Designer), *Tester, Writer.*

2) Realizar a estratégia Exploratório 360º;
3) Definir como a metodologia será aplicada;
4) Construir o plano inicial do projeto.

b) Uma série de dois ou mais ciclos de entrega

O ciclo de entrega possui três ou quatro partes.

1) Fazer uma reavaliação do plano de entregas;
2) Uma ou mais iterações com código testado e integrado;
3) Entrega real para usuários;
4) Realizar um ritual de conclusão, incluindo reflexão sobre o processo e sobre o produto que está sendo criado.

c) Conclusão do ritual: reflexão sobre a entrega

Neste ritual deve-se, além de realizar uma reflexão sobre o projeto, o processo e deixar claro o que funcionou bem e o que não funcionou. A equipe deve ter um momento feliz no final do projeto, uma espécie de comemoração.

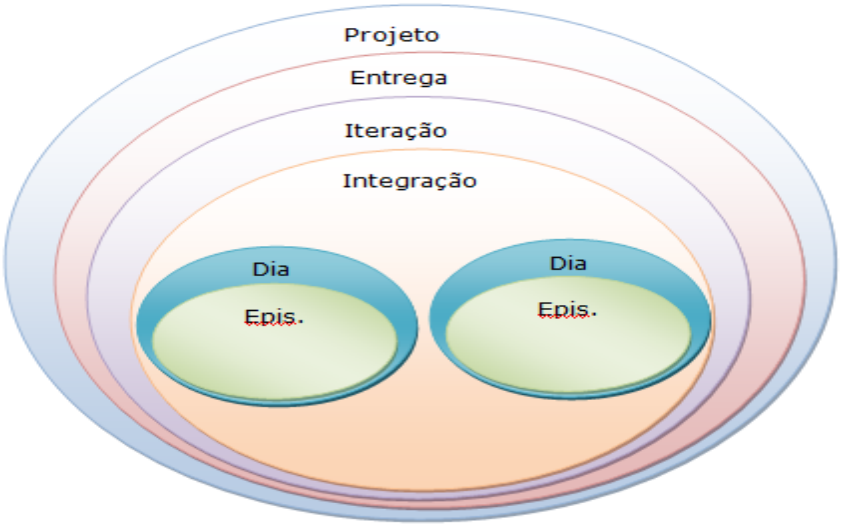

Figure 4. Demonstração do processo.

6. Conclusão

A partir da análise do trabalho conclui-se que a Metodologia *Crystal Clear* é feita para pequenas equipes, permitindo grande flexibilidade em sua aplicação, sendo que o time é fator decisivo da maneira de como ela deve ser aplicada e quais estratégias e técnicas serão adotadas.

Devido a sua flexibilidade, ela não se destina a empresas que desejam ter algo padronizado, uma vez que exige a mudança e adaptação contínua de sua aplicação.

Crystal Clear tem muito a contribuir nas organizações pelo fato de ter como foco o lado humano dos processos de desenvolvimento de *software*, outro fato interessante é o fato da família *Crystal* possuir muitos itens parecidos com outras metodologias, seguem abaixo alguns pontos positivos e negativos observados na metodologia:

Pontos Positivos

- Entregas frequentes das etapas do projeto, reduzindo o retrabalho;
- Reduz possíveis falhas de entrega pois o usuário está diretamente envolvido no projeto;

- Maior controle por parte da gestão, que conhece o que está sendo construído durante a fase de desenvolvimento e não somente ao final;
- Proporciona menos especulação e mais visibilidade das tarefas que vão sendo executadas;
- Possibilita ser muito adaptada de acordo com o projeto.

Pontos Negativos

- A metodologia não foi desenvolvida para trabalhar com projetos longos.
- Até o presente momento não foi bem difundida no Brasil.

7. Referências

Agile Manifesto. (2001) Manifesto for Agile Software Development. Disponível em: < http://agilemanifesto.org >. Acesso em: 15 de abril de 2013.

Cockburn, A. (2004) Crystal Clear: a Human Powered Methodology for Small Teams. Addison Wesley.

Filho, H. F. B. P. (2011) Um estudo analítico entre as abordagens de Engenharia de Requisitos nas Metodologias Ágeis XP, SCRUM e Crystal. Recife: Centro de Informática, Universidade Federal de Pernambuco. Pós-Graduação.

Günal, V. (2012) Agile Software Development Approaches and Their History.

Highsmith, J. (2002) Agile Software Development Ecosystems. Addison Wesley.

Libardi, P. L. O.; Barbosa, V. (2010) Métodos Ágeis. Limeira: Faculdade de Tecnologia, Universidade Estadual de Campinas. Pós-Graduação.

Tomás, M. R. S. (2009) Métodos ágeis: características, pontos fortes e fracos e possibilidades de aplicação. Monte de Caparica: Faculdade de Ciências e Tecnologia, Universidade Nova de Lisboa. Mestrado.