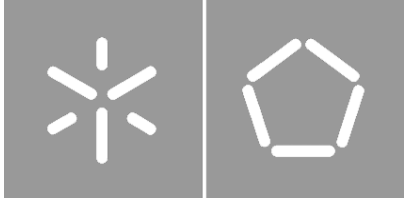
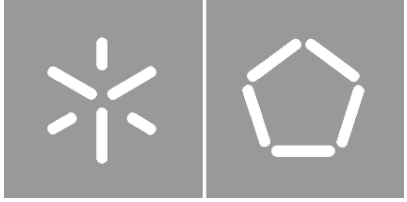


Universidade do Minho
Escola de Engenharia

Marcelo Silva Pereira

Proposta de um sistema de apoio ao processo de gestão de projetos de software numa empresa de Manaus

Março de 2013



Universidade do Minho
Escola de Engenharia

Marcelo Silva Pereira

Proposta de um sistema de apoio ao processo de gestão de projetos de software numa empresa de Manaus

Dissertação
Mestrado em Engenharia Industrial

Trabalho efetuado sob orientação do
Professor Rui M. Lima

Março de 2013

Declaração

Nome: Marcelo Silva Pereira

Endereço eletrónico: marcelo.silvapereira@gmail.com

Telefone: +5592 81130647 / +5592 92125426

Passaporte:

Título dissertação de projeto:

Proposta de um sistema de apoio ao processo de gestão de projetos de software numa empresa de Manaus.

Orientador: Professor Rui M. Lima

Ano de conclusão: 2013

Mestrado em Engenharia Industrial

Universidade do Minho, ___/___/_____

Assinatura:_____

Agradecimentos

Primeiramente a Deus, por me conceder saúde e sabedoria para estar aqui nesse momento. Ao meu pai Sr° Lázaro que sempre me obrigou a tirar nota 110 na escola, querendo mostrar que eu sempre podia ir mais além. À minha mãe Dona Inalda em quem eu sempre me inspirei para estudar desde criança. Ao meu amigo Carlos Vamberto que sempre esteve disposto a me orientar nas dúvidas referente ao desenvolvimento das funcionalidades necessárias. Por fim ao professor Rui Lima pela ótima orientação ao longo destes anos de estudos. A todos muito obrigado.

PROPOSTA DE UM SISTEMA DE APOIO AO PROCESSO DE GESTÃO DE PROJETOS DE SOFTWARE NUMA EMPRESA DE MANAUS

Resumo

Este estudo aborda as fases do desenvolvimento de sistemas desde a listagem dos requisitos do cliente, passando por sua modelagem, desenvolvimento da arquitetura, construção do modelo de dados, camada de persistência, negócio, interface com o utilizador, teste e implantação, alinhada às práticas de gestão de projetos com auxílio de uma ferramenta para gestão de projetos de software (SGPS). Esta ferramenta de software tem como objetivo apontar prazos, custos previstos e realizados ao longo do processo de desenvolvimento de software, o caso de uso desta análise refere-se a um projeto de software de uma empresa de desenvolvimento situado na cidade de Manaus - AM. Neste trabalho apresentamos a ferramenta e análise do caso de uso para o fim de auxiliar na tomada de decisão do gestor de projetos quanto a mudanças no escopo em desenvolvimento.

Palavras-chave: Projecto; Gestão de Projetos; Desenvolvimento de sistemas;

A SYSTEM TO SUPPORT THE PROCESS OF SOFTWARE PROJECT MANAGEMENT DEVELOPMENT IN A BUSINESS COMPANY AT MANAUS

Abstract

This work presents a study about the phases of software development, including customer requirements analysis, software modelling, development, building of data model, persistence layer, business modelling, user interface, test and implementation. These phases were used to develop a tool to support project management practice. This software tool aims to point out deadlines, anticipated and real costs throughout the software development process. This tool was created to support software projects from a development company located in the city of Manaus - AM. In this paper we present the analysis tool and a specific use case from this company, showing that it allows an improved process of decision making along the project, managing scope changes during the development phase.

Key words : Project; Project Management; System Development;

Índice geral

Índice de figuras

1 INTRODUÇÃO

1.1 Enquadramento do Estudo

A eficiência de empresas e/ou institutos de desenvolvimento de sistemas de software é medida pela satisfação que seus produtos oferecem aos seus clientes e facilidades aos seus utilizadores. Quanto mais um sistema informático agrega valor ao processo de negócio de uma empresa, mais o produto de software desenvolvido é necessário para o meio corporativo.

Para que o sucesso de um sistema informático seja pleno, o mesmo deve atender a uma série de requisitos alinhados ao processo de negócio que apoia. Entre muitos requisitos do sistema pode referir-se os seguintes: Facilidades de acesso; retornos confiáveis de informações; apoiar a tomada de decisões; controle das rotinas desenvolvidas no processo de negócio.

No entanto, atingir elevados níveis de satisfação no desenvolvimento de um sistema não é uma tarefa garantida se alguns pontos não forem cumpridos como descritos a seguir: a definição do âmbito em que o sistema informático será inserido, bem como seus limites de escopo e fonteiras de utilização, pelo engenheiro de sistemas; o conhecimento do processo padrão de trabalho por parte do utilizador e as saídas que o mesmo produz para a próxima fase do processo em questão; os dados que devem ser mantidos por parte do utilizador no sistema informático; e os índices que devem ser visualizados para apoio a tomada de decisão por parte do gestor do processo. Estes são apenas alguns dos fatores importantes a esclarecer para o conhecimento do ambiente do processo de negócio que o sistema deve auxiliar.

Tais fatores influenciam de forma significativa o processo de desenvolvimento do sistema, pois sem eles o sistema não faz sentido e consequentemente não tem porque existir, porém, partir para a especificação desse âmbito requer uma atuação cuidadosa de todos que fazem parte desse processo, para se definir os limites do que se deve e do que não se deve atender através do sistema.

Utilizadores, gestores, engenheiros de sistemas e arquitetos de software são figuras que compõem a equipe de definição dos limites do sistema informático, especificando e modelando requisitos, homologando interfaces, definindo tecnologias de desenvolvimento e por fim, os prazos e os custos para a entrega do sistema.

Patterson and Hennessy (2000) afirmam que muitos projetos em grandes companhias foram cancelados por terem ultrapassado o prazo inicialmente previsto para entrega do produto. Tal situação ocorreu pelo fato do gestor das equipes de desenvolvimento não terem acesso a ferramentas com previsões aproximadas de prazos e que permitam gerir a execução do projeto acompanhando cada fase de seu desenvolvimento.

Portanto, encontrar uma alternativa dinâmica que demonstre o processo de produção de sistemas de software mais próximo da sua real situação, desde as atividades realizadas quanto à disponibilidade das equipes, é necessário para que aumente a fiabilidade da empresa de desenvolvimento e consequentemente permitindo o crescimento da sua marca de eficiência no segmento.

1.2 Objetivos do estudo

A partir da metodologia utilizada, este estudo pretende desenvolver um módulo de software que auxilie o processo de produção de sistemas informáticos mostrando o andamento de suas atividades, custos e prazos apoiando a tomada de decisão para eventuais mudanças no projeto por parte do gestor.

1.3 Estrutura da Dissertação

Este trabalho de investigação se inicia com seu enquadramento e apresentação dos objetivos, seguida da pesquisa bibliográfica consultada. Logo depois a metodologia utilizada no projeto de investigação, seguida pela análise dos resultados do projeto, apresentação do produto de software desenvolvido e finalizando com a conclusão, referências bibliográficas e anexos.

2 A GESTÃO DE PROCESSOS DE DESENVOLVIMENTO DE PROJETOS DE SOFTWARE

2.1 O Processo de Desenvolvimento de Software

A princípio o processo de construção de software se dá a partir das fases de desenvolvimento estabelecidas por uma organização. Essas fases de desenvolvimento são importantes para se conhecer os passos necessários para a implementação de um sistema informático desde as demandas específicas junto ao utilizador, a metodologia de desenvolvimento a utilizar, a arquitetura que o sistema irá utilizar, o seu desenvolvimento, modelos de testes e implantação. Para além destes aspetos, Ainda é necessário garantir que o acompanhamento das atividades ao longo do processo de construção do sistema esteja alinhado com as práticas de gestão de projetos, poís é através deste acompanhamento que o gestor de projeto fornece informações sobre o desenvolvimento do mesmo às partes interessadas e visualiza os impactos de possíveis mudanças no decorrer do seu andamento.

2.1.1 O mercado de desenvolvimento de software

Inicialmente há a necessidade de se organizar o modo de produção de um sistema de software. Para isso, deve-se adotar uma metodologia de desenvolvimento que esteja mais adequada às características de trabalho da equipa de trabalho como também as características do produto de software a ser entregue ao final do projeto.

Para facilitar a organização do processo de produção de um sistema informático, várias metodologias de desenvolvimento foram criadas com o objetivo de aperfeiçoar as fases de construção de um sistema. Basta apenas que o gestor de desenvolvimento, levando em consideração os recursos disponíveis, adote uma metodologia que mais se adeque à realidade ou que possa ser utilizada como propulsora das melhores práticas de desenvolvimento da sua equipe.

Essa organização ainda se tem tornado mais necessária devido ao crescimento do mercado de sistemas informáticos direcionado a soluções corporativas (Mertz, 2007), que tinha em 2007 uma projeção de crescimento de 22,1% até 2011 ficando em torno de US $12,3 bilhões, superando os 9% de crescimento esperado do mercado de tecnologia da informação como um todo. Contudo, o mercado mundial de sistemas tinha a previsão de atingir cerca de US $ 14,5 bilhões em 2012 e de acordo com os relatórios trimestrais de acompanhamento da Gartner, Inc a projeção é de que até 2015 esse mercado deve atingir a receita de US $22,1 bilhões (Mertz, 2012). Este crescimento tem sido maior do que o esperado devido ao avanço das tecnologias móveis, ferramentas de desenvolvimento e necessidade das empresas possuírem sistemas de software que apoiem a gestão de seus negócios.

“Em 2011, 63% dos produtos do mercado de sistemas informáticos como serviço vai suportar web service e tecnologias de web” (Mertz, 2007), promovendo uma maior operacionalização dos negócios da empresa a partir de qualquer parte do mundo através da internet. Tendendo a partir de então que todos os sistemas a serem construidos tenham essa arquitetura. Esta arquitetura viabiliza o acesso direto por parte do cliente de qualquer lugar do mundo com acesso a rede mundial de computadores, possibilitando ao utilizador acesso à informação da sua empresa como se o mesmo nunca tivesse saído dela. Mertz (2007) ainda afirma que o sistema informático como serviço via Web é mais do que uma opção hoje: “É um elemento que está ganhando cada vez mais importância no mercado com o passar do tempo”, principalmente para as empresa divulgarem suas marcas mundo a fora (visibilidade) e consequentemente atraírem novos negócios.

Porém existem obstáculos para o avanço dos sistemas informáticos direcionados a gestão de empresas. A maioria das empresas de desenvolvimento de sistemas está reestruturando os seus produtos de software para a arquitetura orientada a serviços, ou seja, aplicações de software com requisitos muito abrangentes direcionados a propósito gerais de serviço onde contém muitas funcionalidades que por sua vez não são 100% necessária aos seus utilizadores, exigindo alto valor de investimento, extenso prazo de desenvolvimento, o que obriga seus clientes a investirem na migração de seus recursos tecnológicos para essa arquitetura.

Por outro lado esses clientes têm a oportunidade de avaliar se os sistemas informáticos como serviços são alternativas válidas para outros aspetos do negócio, quer dizer, sistemas mais específicos para as suas necessidades de negócio, e que são desenvolvidos em menos tempo e com menor custo (Mertz, 2007).

No entanto, desenvolver sistemas de software em curto período de tempo, direcionado a aplicações corporativas não é fácil, devido, entre outros aspectos, à consideração das necessidades reais de cada cliente, ao grande número de fases que constam nos processos de negócio, e às regras de negócios complexas e específicas de cada empresa. Cada cliente possui uma necessidade específica de acordo com os seus processos de negócio, o que obriga empresas de desenvolvimento de sistemas a passar muito tempo na fase de planeamento do projeto com o objetivo de entender o funcionamento real do processo de negócio do cliente.

Blaschek (2009) cita uma série de fatores que contribuem para a inviabilidade do desenvolvimento da grande maioria dos sistemas de software: várias mudanças nos requisitos do utilizador no decorrer do projeto; prazos de entrega e custos de investimento estimado de maneira errada; conflitos frequentes entre colaboradores da equipe de desenvolvimento; utilizadores insatisfeitos com o produto apresentado; equipes desmotivadas com baixo nível de produção. Estes são alguns fatores que frequentemente são encontrados em ambientes de desenvolvimento de projetos de software,

contribuindo para a perda dos investimentos e consequentemente prejuízos para toda a empresa de desenvolvimento.

Para contornar essa situação o uso das práticas de gerenciamento de projetos, especificamente, propostos pela literatura, traz uma enorme contribuição para melhorar esse problema. Para torná-lo ainda mais eficiente, é necessário alinhar essas práticas com a engenharia de software.

Tornar mais eficiente o processo de desenvolvimento de sistemas informáticos, num mercado que é totalmente dinâmico e imprevisível, através de métodos e técnicas específicas de gestão de projetos auxiliadas por metodologias de engenharia de software deve disponibilizar soluções rápidas e consistentes (produtos de software) aos clientes proporcionando a empresa de desenvolvimento maior competitividade de mercado e consequentemente crescimento dos seus negócios.

O dinamismo das necessidades de software do cliente faz com que empresas de desenvolvimento tentem se adequar e buscar metodologias que proporcionem respostas rápidas e baixo custo em construção e aperfeiçoamento de sistemas para os mesmos, melhorando o processo de produção de sistemas e proporcionando a empresa de desenvolvimento responder com maior agilidade e eficiência as necessidades do mercado de sistemas junto aos seus clientes.

2.1.2 A engenharia de software

Engenharia de software é uma área do conhecimento da computação voltada para a especificação, desenvolvimento e manutenção de sistemas de software aplicando tecnologias e práticas de gerência de projetos, objetivando organização, produtividade e qualidade. É o arcabouço que constitui o processo, os conjuntos de métodos e ferramentas necessárias (Pressman, 2009). Esta área de conhecimento surgiu com o objetivo de utilizar princípios de engenharia no desenvolvimento de software para aumentar a qualidade dos produtos oferecidos, diminuindo custos e riscos relacionados e criar processos repetíveis e eficazes para serem utilizados nos ciclos de manutenção e desenvolvimento de software, que com o passar dos anos foram se tornando modelos a serem seguidos por outros ambientes de desenvolvimento. Por sua vez outros ambientes desenvolveram seus próprios modelos para se adaptarem melhor à sua realidade.

Uma metodologia de processo de desenvolvimento de software, ou simplesmente modelo de processo, pode ser visto como uma representação ou abstração dos objetos e atividades envolvidas no processo de software (negócio inserido). Além disso, oferece uma forma mais abrangente e fácil de representar a gestão do processo de desenvolvimento de software e consequentemente do progresso do projeto. A Tabela 1 apresenta as vantagens e desvantagens de alguns modelos atualmente conhecidos.

Tabela 1 - Relação entre modelos atuais de engenharia de software.

Modelo	Característica	Vantagem	Desvantagem
Sequencial ou cascata	Apresenta fases distintas de especificação, projeto e desenvolvimento.	• Torna o processo de desenvolvimento estruturado; • Tem uma ordem sequencial de fases; • Cada fase cai em cascata na próxima e cada fase deve estar terminada antes do início da seguinte; • Todas as atividades identificadas nas fases do modelo são fundamentais e estão na ordem certa;	• Não fornece feedback entre as fases e não permite a atualização ou redefinição das fases anteriores; • Não suporta modificações nos requisitos; • Não prevê a manutenção; • Não permite a reutilização; • É excessivamente sincronizado; • Se ocorrer um atraso todo o processo é afetado; • Faz aparecer o software muito tarde;
Desenvolvimento interativo e incremental	O modelo iterativo e incremental é uma alternativa para solução de problemas enfrentados no modelo cascata. Nesse modelo, considera-se o desenvolvimento do *software* em ciclos iterativos onde uma pequena porção dos requisitos passa por todas as etapas de desenvolvimento como em um modelo cascata. Ao final de cada ciclo de iteração tem-se uma nova versão do *software*, a qual será incrementada a cada novo ciclo que ocorrer. Isso reduz a ansiedade do utilizador em relação ao *software*.	• Menor custo e menos tempo são necessários para se entregar a primeira versão. • Riscos associados ao desenvolvimento de incrementos são menores, devido ao seu tamanho reduzido. • Número de mudanças nos requisitos pode diminuir devido ao curto tempo de desenvolvimento da primeira versão.	• Se os requisitos não são tão estáveis ou completos quanto se esperava, alguns incrementos podem precisar ser retirados de uso e retrabalhados ; • A gestão de custo, cronograma e configuração é mais complexo.
Evolucional ou prototipação	Especificação, projeto e desenvolvimento de protótipos.	• Melhora a qualidade da especificação do software, contribuindo para uma queda nos custos de desenvolvimento e manutenção. • Antecipa o treinamento dos utilizadores. • Partes do protótipo podem ser aproveitadas no desenvolvimento do sistema.	• O custo na maioria dos casos é considerado muito alto. • O cliente tende a confundir o protótipo com uma versão do sistema.
Espiral	Evolução através de vários ciclos completos de especificação, projeto e desenvolvimento.	• Suporta mecanismos de redução de risco • Inclui interações • Reflete as práticas reais da engenharia atual • Apresenta uma abordagem sistemática • Estimativas tornam-se mais realísticas com o progresso do trabalho, porque problemas importantes são descobertos mais cedo. • É mais versátil para lidar com mudanças que desenvolvim. de software geralmente exige.	• Avaliação dos riscos exige muita experiência. • O modelo é relativamente novo e não tem sido amplamente utilizado

A Figura 1 representa o modelo de desenvolvimento em cascata que consiste em vários processos estruturados de forma linear que precisam estar totalmente concluídos para o início da próxima atividade na sequência. O modelo permanece restrito em uma única sequência do início ao fim.

Figura 1 – Representação do modelo de desenvolvimento em cascata

Na Figura 2 podemos observar que o projeto principal é dividido em módulos menores (incrementos) e a partir de então desenvolvido em partes. O desenvolvimento por partes é semelhante ao modelo cascata, porém partes maiores do projeto de software como estudo e projeto do mesmo, já foram definidos anteriormente.

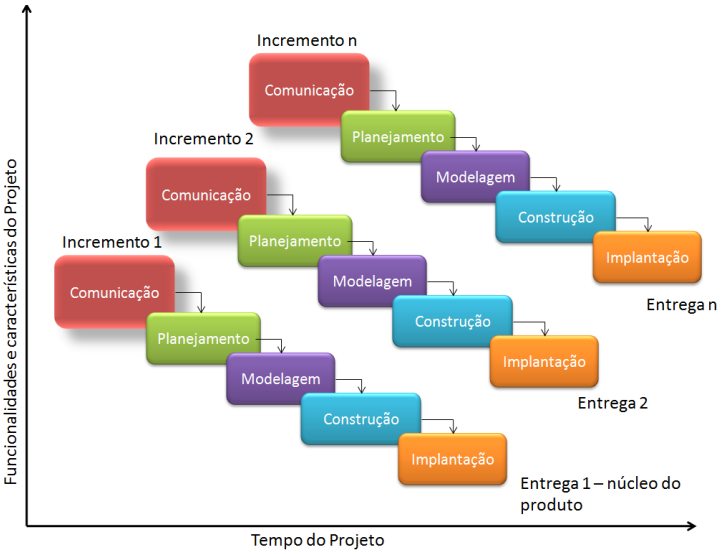


Figura 2 – Representação do modelo de desenvolvimento iterativo e incremental (Pressman, 2009)

A Figura 3 sugere o sistema ser desenvolvido em parte de um esboço que é documentado, desenvolvido e validado junto ao cliente ao mesmo tempo. O risco que este modelo propõe é que se o protótipo não atender aos requisitos do cliente o esforço gasto desenvolvendo o protótipo é totalmente perdido, tento então que partir para uma nova abordagem. Por outro lado, se a prototipação atender as necessidades do cliente, juntamente com a documentação aprovada, o sistema fica mais próximo de sua conclusão.

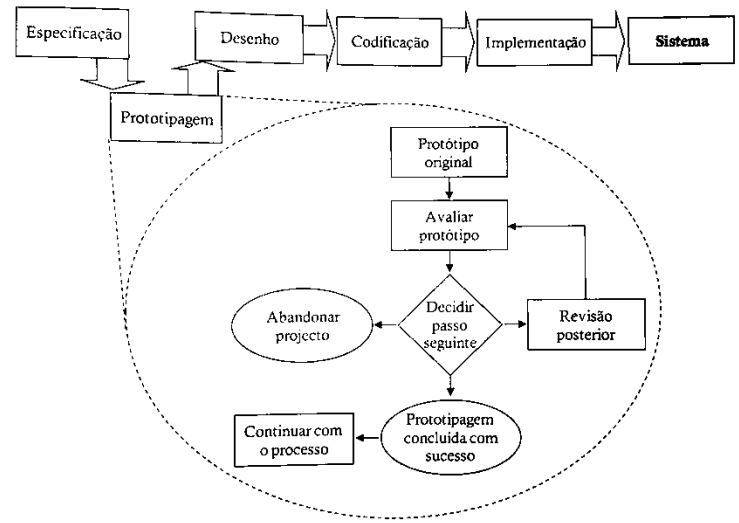


Figura 3 – Representação do modelo de desenvolvimento evolucional ou prototipação (Miguel, 2010)

Por fim, a Figura 4 aborda a continuidade do ciclo do processo de desenvolvimento definindo seus objetivos, analisando os ricos, desenvolvendo, testando, implantando e planejando a próxima fase do projeto, no entanto esse modelo exige da equipe de desenvolvimento muita maturidade dos processos de construção o que diminui devido às inúmeras variáveis que compõem os sistemas a serem desenvolvidos.

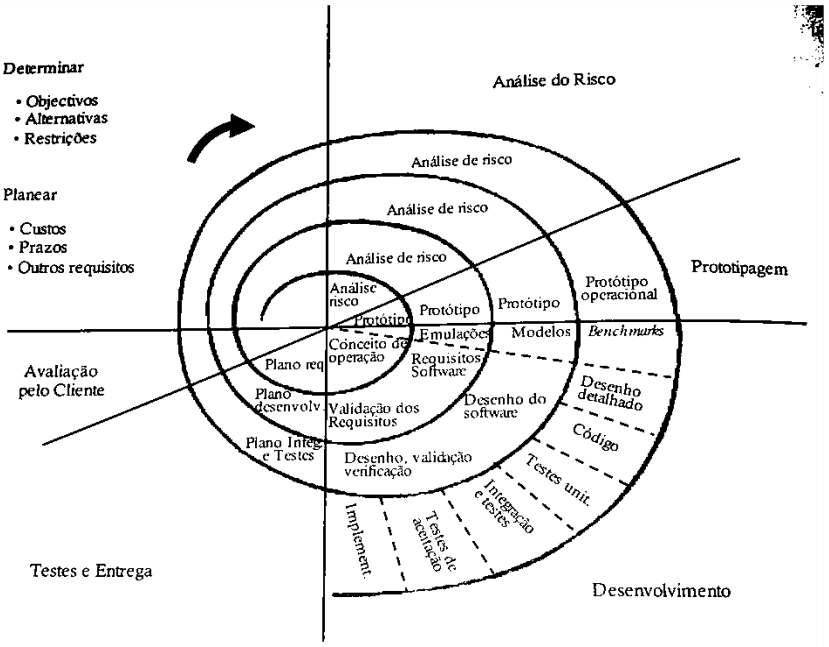


Figura 4 – Representação do modelo de desenvolvimento espiral (Miguel, 2010)

2.1.3 Metodologias ágeis

Alinhada a engenharia de software que define a metodologia, os passos a serem seguidos, os procedimentos a serem utilizados pela equipa de desenvolvimento, as metodologias ágeis propõem uma gestão das atividades objetivando a entrega mais ágil do produto em desenvolvimento.

Entre inúmeras metodologias ágeis, destaca-se atualmente a metodologia Scrum que de acordo com Kniberg (2007) é um framework de boas práticas de gestão de atividades a serem desenvolvidas, ou seja, ele não diz como fazer, mas sim o que se deve fazer.

A principal característica do Scrum é o product backlog, que contém todas as necessidades dos clientes, no entanto a equipe de desenvolvimento, de acordo com a prioridade dos desejos do cliente, estima a quantidade de esforço necessário para o desenvolvimento de uma funcionalidade que deverá ser entregue finalizada ao final de um sprint. A Figura 5 ilustra a realização de um ciclo de trabalho de uma equipe de desenvolvimento utilizando Scrum.

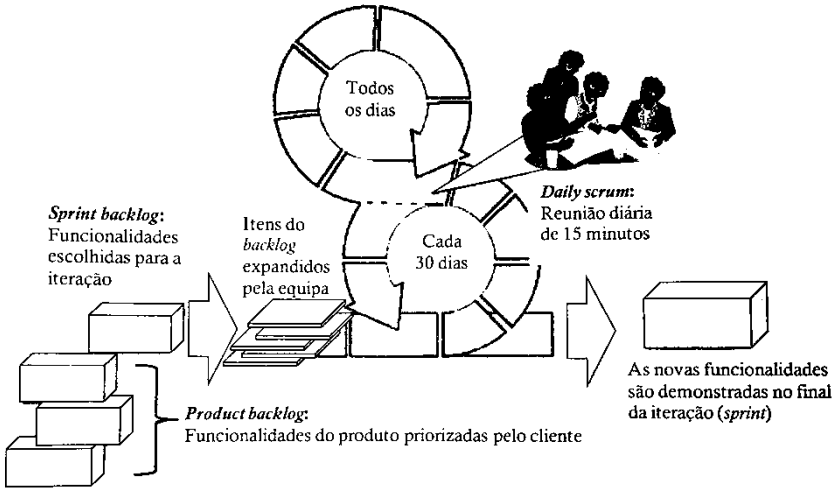


Figura 5 – Ciclo de iteração Scrum (Miguel, 2010).

2.1.4 Fases de processo de desenvolvimento de sistemas

Como qualquer produto a ser produzido, todo sistema de software deve obedecer a fases de construção que visam garantir que todos os requisitos funcionais solicitados pelo cliente sejam desenvolvidos na aplicação e entregues com um nível satisfatório de aceitação.

No entanto com o dinamismo dos negócios dos clientes, novas necessidades de controlos dos processos de negócio, atualizações do sistema de software, faz-se necessário o desenvolvimento de novas funcionalidades a serem incluídas nas aplicações atuais devido às correntes mudanças. Consequentemente, desenvolver de forma rápida as novas rotinas necessárias torna-se um diferencial para as empresas de desenvolvimento de sistemas que conseguem em um baixo custo e tempo de resposta adequar os sistemas atuais as novas regras de negócios solicitadas e/ou impostas.

Com isso, algumas empresas acabam simplificando seus processos de desenvolvimento de sistemas de modo a torna-los mais enxutos e menos demorados. Algumas empresas de desenvolvimento por sua vez aderem às metodologias de desenvolvimento ágeis que tem o foco maior em atender e satisfazer as necessidades do cliente, sem aderir ao processo de desenvolvimento tradicional de sistemas. No entanto a seguir são apresentados tópicos básicos para o desenvolvimento de um sistema bem como suas ferramentas.

2.1.5 Análise de Negócio e Modelação de Processos

A análise de negócio, como a primeira fase do processo de desenvolvimento de software, consiste exatamente em realizar a primeira abordagem junto ao cliente para entender suas reais necessidades e avaliar se a nova funcionalidade do sistema realmente atenderá as expectativas do cliente para o melhor andamento e controle dos seus processos. Muitas empresas, até as que utilizam metodologias

de desenvolvimento ágil estão adotando essa fase em seus processos de desenvolvimento justamente para avaliar se as funcionalidades a serem desenvolvidas realmente irão agregar o valor desejado aos seus clientes.

Nessa fase, o analista de negócio, compõe junto ao cliente o documento de visão, o modelo de processo e as propostas de interface de utilizador que a funcionalidade deve atender.

O *documento de visão* descreve exatamente o funcionamento da funcionalidade desejada, suas ações, iterações e controles de maneira textual clara e objetiva, proporcionando a visão necessária aos analistas de sistemas, desenvolvedores e analistas de teste sobre o funcionamento e os impactos proporcionados pela mesma ao sistema e ao negócio do cliente.

A *interface de utilizador* propõe um modelo de interface que atenda as necessidades do utilizador e descreva na mesma as funcionalidades disponíveis, a disposição dos dados a visualizar e a navegabilidade do sistema de forma intuitiva para o cliente. A Figura 6 exibe um exemplo de interface do utilizador.

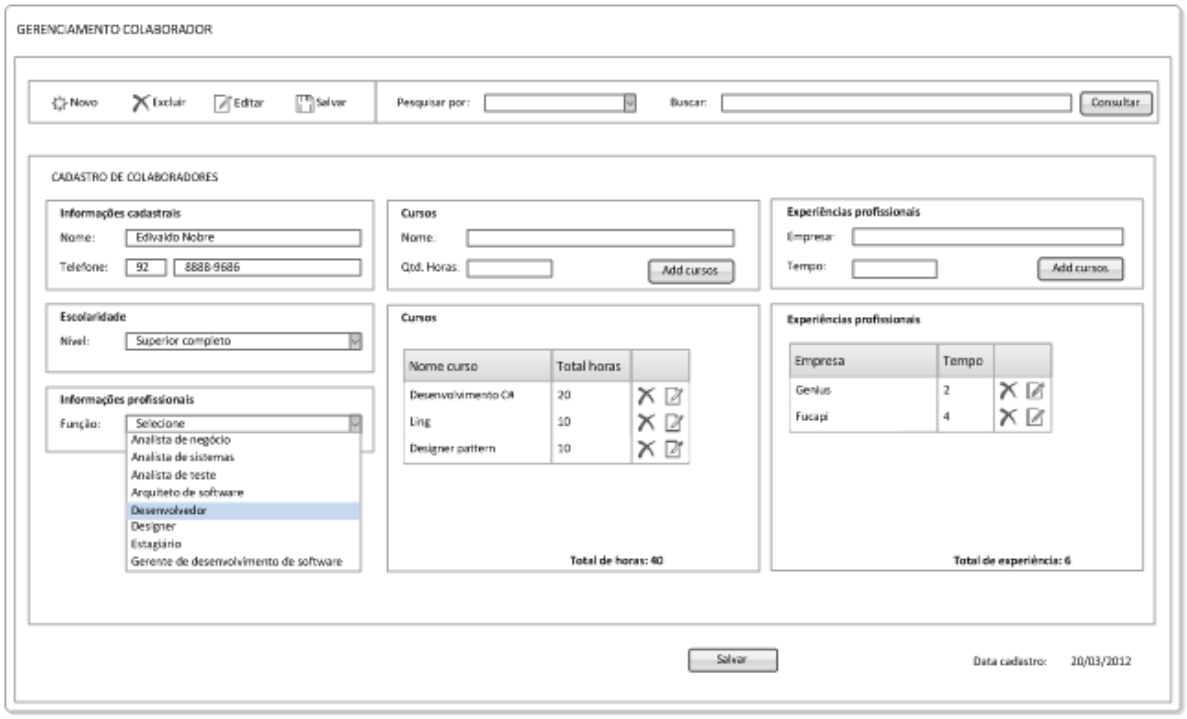


Figura 6 – Exemplo de layout de cadastro de colaborador

O *modelo de processo* apresenta através de diagrama de fluxo a iteração do sistema com todos os setores interessados e colaboradores a utilizar a rotina em seu respetivo processo de negócio de forma objetiva, inclusive suas respetivas tomadas de decisão.

Inúmeras linguagens auxiliam na elaboração do fluxo de negócio de uma empresa. Várias descrevem eficientemente os complexos níveis de detalhes que compreendem o processo de negócio, porém não disponibiliza a visibilidade necessária para que a implantação de um projeto tenha sucesso (Bortolini, 2006). O BPMI.org (Business Process Management Initiative), preocupada em gerar uma nova e atualizada especificação para descrever processos de negócios, lançou em 2002 o BPML – Business Process Modelling Language – que além de descrever o processo de negócio, expõe de maneira objetiva a execução de uma rotina e permite representar o processo de negócio de uma organização como um todo. A utilização do BPML na modelagem do processo para a implantação de um projeto auxilia na compreensão do processo como um todo, disponibilizando aos colaboradores o impacto que cada fase de processo pode gerar (Hoyer, Bucherer & Schnabel, 2007).

A aplicação da linguagem BPML para descrição dos processos a serem desenvolvidos numa empresa de desenvolvimento de software, contribui para a diminuição do tempo que o engenheiro de sistemas necessita para estudar e abstrair cada fase do processo de negócio para o desenvolvimento do modelo de software, relativamente à descrição em ferramentas que não unificam os processos. A redução no tempo de modelação dos processos torna a fase de descrição dos requisitos de sistemas mais rápida e simples, reduzindo também o tempo de construção do software.

A linguagem apresenta de forma tão clara e genérica as fases que consistem o processo de negócio que pode ser adotada por outras áreas que necessitam da visibilidade das rotinas que antecedem uma determinada fase do processo. Como exemplo a área de controlo e gestão da qualidade pode utilizar o mesmo diagrama BPML para mapear os processos de negócio da empresa e implantar políticas de melhorias de processo em suas respetivas fases.

Uma das maiores vantagens da linguagem é representar de forma simples os dados necessários para a execução do processo, o funcionamento do processo principal e o resultado final do processo que pode servir como entrada para um novo processo.

2.1.6 Análise do sistema

Um sistema não pode ser desenvolvido antes de se saber realmente que necessidades de facto precisa satisfazer. A análise do sistema propõe especificar os requisitos necessários para o funcionamento pleno e eficiente do sistema a ser desenvolvido.

O levantamento e a especificação dos requisitos

É comum afirmar que a fase mais importante e de maior custo para o projeto de desenvolvimento de software é a sua fase de construção. Porém o que muitos não sabem é que na fase de construção todos os limites do projeto já estão delimitados, todo o âmbito está definido, todo requisito já está

detalhado e homologado com o utilizador e todas as suas funcionalidades estão representadas nos seus respetivos modelos (diagramas) de funcionamento.

Porém, a definição de todas as funcionalidades do sistema, expectativas e necessidades do utilizador devem ser detalhadas e homologadas antes da fase de construção do sistema através das especificações de requisitos e formalizadas em documentos de especificação de requisitos.

Os requisitos do sistema são divididos em funcionais e não funcionais. O primeiro define as funcionalidades ou ações que o sistema deve disponibilizar aos utilizadores na sua entrega final. Já os requisitos não funcionais descrevem atributos do sistema ou do ambiente do mesmo como: extensibilidade, usabilidade, confiabilidade, desempenho, escalabilidade, reusabilidade, capacidade de manutenção, reutilização de código, desempenho, eficiência no desenvolvimento e confiabilidade nos dados apresentados (Engholm, 2010).

Nessa fase é realizada a *listagem dos requisitos* ou *levantamentos dos requisitos*. É a partir desta atividade que se inicia o processo de descrever detalhadamente todos os objetivos e restrições envolvidas não só para o correto planeamento, viabilidade e custo, mas também para produzir um sistema que atenda as necessidades e expectativas do utilizador. Logo o entendimento e controlo desses requisitos são fundamentais para o sucesso do projeto e seus objetivos.

No entanto, um dos grandes obstáculos no levantamento de requisitos é quando ele deve ser realizado em ambientes no qual as suas fontes de informação (utilizadores) não disponibilizam tempo para reuniões de levantamento/especificações ou não disponibilizam arquivos que esclareçam os objetivos do projeto a desenvolver.

Porém, listar requisitos deve ser entendido como um processo de descoberta e entendimento do domínio do problema a ser atendido pelo projeto de software a ser desenvolvido, além das necessidades de negócio a serem auxiliadas (Engholm, 2010). Essa fase é executada pelo engenheiro de sistemas e os utilizadores a serem atendidos pelo projeto.

As informações são listadas por meio de reuniões, entrevistas, análises de documentos, análises de sistemas já utilizados, observações de tarefas realizadas no dia-dia da empresa, diagramas de workflow e são alinhados aos problemas de negócio a ser atendido pelo projeto de software e registrado em ata todos os requisitos acordados.

Todos os procedimentos a serem adotados no processo de levantamento de requisitos devem estar registrados no documento de especificação de requisitos, que por sua vez descreve em detalhes todos os procedimentos a serem adotados para o desenvolvimento de determinada funcionalidade, como listado abaixo:

- *Descrição do documento de caso de uso*: Descreve a funcionalidade principal a ser especificada pelo documento em questão. É responsável por identificar a funcionalidade que está se detalhando.
- *Layout sugerido da funcionalidade:* Exibe o layout de interação com o utilizador para a utilização da funcionalidade descrita pelo documento de caso de uso. Este layout deve ser previamente aprovado junto ao utilizador nas reuniões de especificação de requisitos.
- *Relacionamento com outros casos de uso:* Informa se esta funcionalidade possui algum relacionamento com outras funcionalidades descritas em outros documentos de casos de uso. Alguns documentos especificam funcionalidades que são apenas partes de um processo, ou processos que suas saídas são entradas para outros processos descritos em outros documentos.
- *Campos adotados na pelo layout sugerido:* Descreve os campos utilizados no layout sugerido, bem como os dados que são recebidos/exibidos, seus tipos de dados e suas regras de exibição.
- *Atores:* Especifica os atores que podem utilizar esta funcionalidade bem como os que possuem acesso a este layout sugerido quando o sistema estiver funcionando.
- *Pré – condições:* Define condições necessárias para que o funcionamento seja efetuado pelo utilizador do sistema, como acesso ao sistema, ter acesso a funcionalidade, hierarquia e/ou acessibilidade do sistema.
- *Pós – condições:* Especifica os resultados disponibilizados gerados pela funcionalidade em questão para as próximas funcionalidades que dependem ou não deste caso de uso.
- *Regras de negócio:* Contendo uma das informações mais importantes para o documento de caso de uso, a regra de negócio descreve como a funcionalidade se comporta com execuções inesperadas, tratamentos de erros e não conformidades com o processo padrão de negócio que o sistema se aplica. Aqui são retratados todos os riscos que podem ocorrer no momento de execução da funcionalidade bem como as rotinas de contorno desses riscos.
- *Fluxo principal:* Descreve de forma textual o funcionamento ideal da funcionalidade descrita no caso de uso elaborado.

Por fim, a modelagem descreve e apresenta seus respetivos diagramas de caso de uso; diagramas de atividades; diagrama de classe; diagrama de sequência e diagrama de estados.

Modelagem do sistema

Um modelo é uma simplificação da realidade. Os modelos podem realizar planos detalhados, assim como planos mais gerais com uma visão panorâmica do sistema. Para modelagem da maioria dos sistemas de software é utilizada a UML (Unified Modeling Language), que permite que o sistema seja descrito sob diferentes aspetos, utilizando modelos específicos para cada caso representando os requisitos solicitados.

Conforme (Engholm, 2010), um requisito é uma condição que o sistema necessita para atingir o seu objetivo. O objetivo de todo sistema é atender a um conjunto de requisitos, as necessidades que o sistema deve satisfazer.

Modelar os requisitos de um sistema simplifica o seu desenvolvimento, norteia um programador. A UML não é um método de desenvolvimento, o que significa que ela não diz o que fazer primeiro e em seguida ou como desenhar o sistema, mas auxilia a visualizar seu desenho e a comunicação entre objetos. Por sua vez é composta por muitos elementos de modelo que representam as diferentes partes de um sistema de software, onde o mesmo busca levantar as necessidades e requisitos para seu desenvolvimento e modelagem para descrevê-los de forma mais intuitiva em interações. Os elementos UML são usados para criar diagramas, que representam uma determinada parte, ou um ponto de vista do sistema. Abaixo estão descritos alguns modelos importantes:

- Diagrama de caso de uso descreve a funcionalidade proposta para um novo sistema, que será projetado. Podemos dizer que um caso de uso é um "documento narrativo que descreve a sequência de eventos de um ator que usa um sistema para completar um processo". Casos de uso são considerados como comportamentos e funcionalidades desejadas para o sistema visíveis aos utilizadores. Comportamento refere-se ao que o sistema deve fazer como resposta a eventos gerados da interação do mesmo com seus utilizadores (Engholm, 2010).
- Diagrama de atividades pode ser visualizado o início, fim, atividades e fluxo alternativos associados ao fluxo de eventos de um caso de uso. Na criação de um diagrama de atividade é possível executar tarefas como de identificar atividades, identificar loops, atividades paralelas e decisões. Segundo Bezerra (2002), diagrama de atividades é um tipo especial de diagrama de estados, onde são apresentados os estados de uma atividade, em vez dos estados de um objeto. Estes diagramas são orientados a fluxos de controlo. Engholm (2010) resume dizendo que um diagrama de atividades permite modelar o comportamento do sistema, denotando os caminhos lógicos que um processo pode seguir.
- *Diagrama de classe* demonstra a estrutura estática das classes de um sistema onde estas representam as "coisas" que são gerenciadas pela aplicação modelada. Classes podem se relacionar com outras através de diversas maneiras: associação (conectadas entre si),

dependência (uma classe depende ou usa outra classe), especialização (uma classe é uma especialização de outra classe), ou em pacotes (classes agrupadas por características similares).

- *Diagrama de sequência* mostra a troca de mensagens entre diversos objetos, em uma situação específica e delimitada no tempo. Coloca ênfase especial na ordem e nos momentos nos quais mensagens para os objetos são enviadas. Os objetos são representados através de linhas verticais tracejadas (denominadas como linha de existência), com o nome do objeto no topo. O eixo do tempo e também vertical, aumentando para baixo, de modo que as mensagens são enviadas de um objeto para outro na forma de setas com a operação e os nomes dos parâmetros. Os diagramas de sequência procuram mostrar a execução de um processo ou tarefa, desde sua criação até sua conclusão, bem como os responsáveis por cada evento e suas respetivas trocas de mensagens.
- *Diagrama de implantação* descreve os componentes de hardware e software e sua interação com outros elementos de suporte ao processamento. Representa à configuração e a arquitetura de um sistema em que estarão ligados seus respetivos componentes, sendo representado pela arquitetura física de hardware, processadores etc.

Modelo Entidade Relacional – ER

O modelo Entidade-Relacionamento (ER, ou também chamado Entidade Associação) é usado na maioria dos métodos e ferramentas de auxílio à conceção de BD's (MERISE, IDA, Yourdon, ERWin). A ideia fundamental deste modelo é a de conservar como conceitos de base os conceitos genéricos (objetos, associação, propriedade) usados no processo de abstração que vai da observação de uma realidade à sua descrição. Existem três elementos fundamentais para a elaboração do modelo entidade relacional: entidade, associação e os atributos. A Figura 7 apresenta um exemplo de um modelo ER.

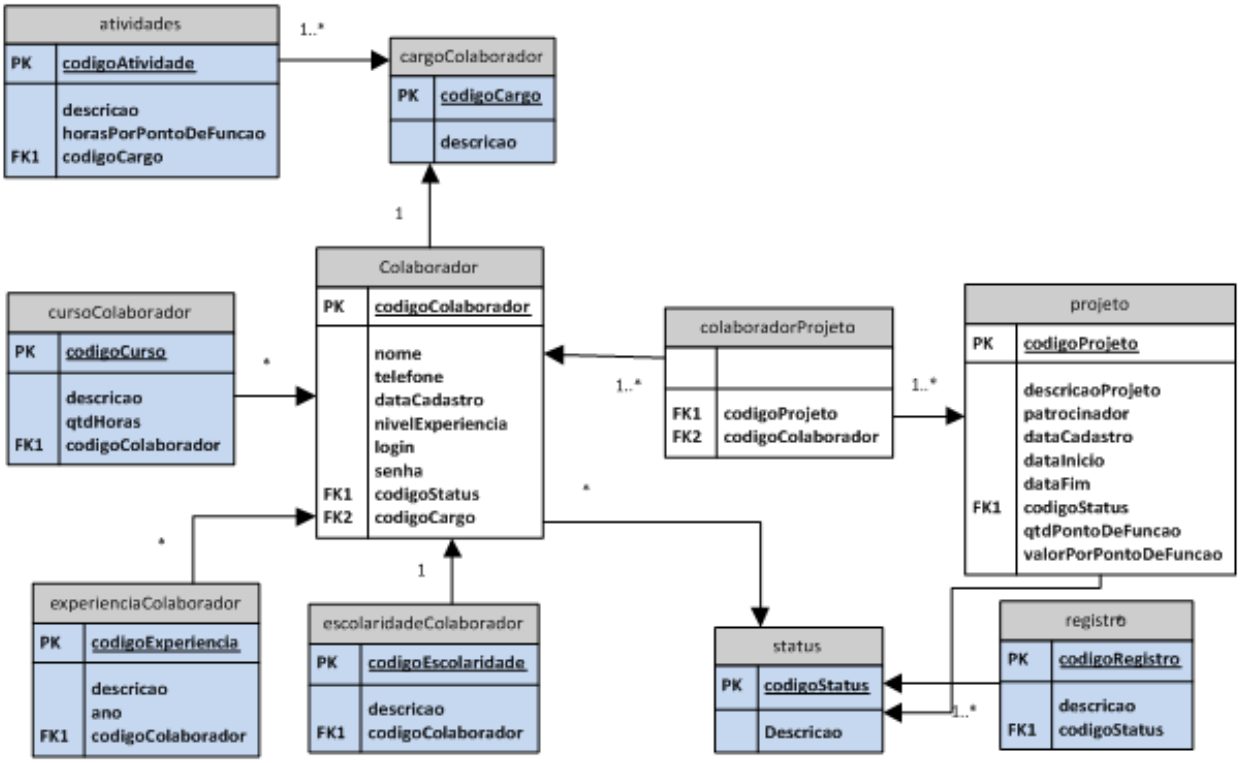


Figura 7 – Modelo de diagrama entidade relacional

2.1.7 Arquitetura do Sistema

Definir bem o ambiente em que se desenvolve é uma decisão importante para o processo de desenvolvimento de sistemas, pois impõem ao cliente do produto adequação ao ambiente necessário para a execução do sistema de software construído. A fase de arquitetura é composta pela execução dos requisitos não funcionais definidos pelo engenheiro de sistemas e aprimorar as ferramentas a fim de diminuir o tempo de desenvolvimento dos requisitos funcionais definindo a linguagem de programação junto com seu ambiente de desenvolvimento, as bibliotecas a utilizar ou a construir (framework) e por fim, a camada onde os dados manipulados pela aplicação a ser desenvolvida irá persistir seus dados definindo assim seu banco de dados. Abaixo estão listados alguns itens importantes para uma arquitetura bem consolidada:

- A *Linguagem de programação*, de acordo com Willrich (2004), tem seu conceito dividido em linguagem de baixo nível e linguagem de alto nível. Mesmo com a evolução dos ambientes de desenvolvimento, incluindo as linguagens, a definição delas continua sendo a mesma.

> "As linguagens de programação, mais especificamente as de alto nível são assim denominadas por apresentarem uma sintaxe mais próxima da linguagem natural, fazendo uso de palavras reservadas extraídas do vocabulário corrente (como READ, WRITE, TYPE, etc...) e permitirem a manipulação dos dados nas mais diversas formas (números inteiros, reais, vetores, listas, etc...);"

Com isso, inúmeras linguagens surgiram de acordo com as necessidades de plataformas de desenvolvimento, onde as mesmas possuem pequenas diferenças de sintaxe, passagem de parâmetros, chamadas de métodos entre outros detalhes percetíveis apenas pelo desenvolvedor do sistema. Abaixo a tabela apresenta algumas características de linguagens de programação utilizadas atualmente.

Tabela 2 - Relação entre linguagens de programação atuais.

Linguagem	Java	PHP	C Sharp
Característica	É uma linguagem de programação orientada a objetos. É antes de tudo uma linguagem simples, robusta, segura, extensível, bem estruturada, distribuída, *multithreaded* e com coletor de lixo.	É uma linguagem orientada a objetos onde ao invés do código ser compilado, ele é interpretado. O PHP é uma linguagem livre e atualmente muito utilizada para gerar conteúdo dinâmico.	É uma linguagem de programação orientada a objetos. A sua sintaxe orientada a objetos foi baseada no C++, mas inclui muitas influências de outras linguagens de programação e Java.
Plataforma de funcionamento	Independente de plataforma	Independente de plataforma	Independente de plataforma
Ambiente de funcionamento	Web e Desktop	Web	Web e Desktop
Desenvolvido por	Sun Microsystems	Livre	Microsoft
Vantagem	Não possui sobrecarga de operadores, structs, unions, aritmética de ponteiros, herança múltipla, arquivos. h, diretivas de pré-processamento e a memória alocada dinamicamente é gerenciada pela própria linguagem, que usa algoritmos de coletor de lixo para liberar regiões de memória que não estão mais em uso.	Velocidade, robustez, estruturação e orientação a objetos, portabilidade e tipagem dinâmica.	Facilidade na sintaxe de programação, métodos prontos pra rotinas, fácil acesso a bancos de dados.

- *Framewok* são nada mais que bibliotecas de funcionalidades prontas que uma vez desenvolvidas podem ser transformadas em componentes e reutilizadas por outros sistemas através de chamadas de seus métodos internos. Com o surgimento do paradigma da orientação a objetos, a tecnologia adequada para reuso de grandes componentes tornou-se disponível e resultou na definição de frameworks orientados a objetos. Os principais benefícios dos frameworks orientados a objetos decorrem da modularidade, reusabilidade, extensibilidade e inversão de controlo que eles oferecem aos desenvolvedores, otimizando o tempo de desenvolvimento de um determinado sistema que possui as mesmas características de outros

já desenvolvidos, consequentemente aumentam modularidade através do encapsulamento de detalhes voláteis de implementação por trás de interfaces estáveis. A modularidade dos frameworks ajuda a aumentar a qualidade do software, concentrando o impacto das mudanças de projeto e implementação, o que reduz o esforço necessário para entender e manter softwares existentes. Frameworks maduros permitem uma redução de mais de 90% do volume de código fonte que tem que ser escrito para desenvolver uma aplicação, quando comparado com software escrito com o suporte de uma biblioteca convencional de funções.

- *Banco de dados* é uma coleção de dados estruturados, organizados e armazenados de forma persistente, proporcionando atualizações automáticas dos dados. Alguns dos benefícios de utilização de um banco de dados é a centralização dos dados, tornando seu acesso mais rápido. A confiabilidade é outro ponto importante, pois julgando os dados sejam reais é possível obter diversas informações referentes a uma organização por exemplo. Alguns modelos de construção de dados para determinadas aplicações possibilitam a maior disposição e velocidade de acesso às informações tornando a aplicação mais confiável e eficiente.

2.1.8 Desenvolvimento

Dentre as diversas maneiras de desenvolvimento de sistemas, abaixo seguem listados uma estrutura de desenvolvimento de sistemas utilizando a tecnologia .net (exemplificada na Figura 8) com modelos de acesso a dados utilizando Linq:

- *.NET* é uma plataforma de software que conecta informações, sistemas, pessoas e dispositivos. Nela conectamos uma grande variedade de tecnologias de uso pessoal, de negócios, de telefonia celular a servidores corporativos, permitindo assim, o acesso rápido a informações importantes onde elas forem necessárias e imprescindíveis. Desenvolvido sobre os padrões de Web Services XML, o framework ".NET" possibilita que sistemas e aplicativos, novos ou já existentes, conectem seus dados e transações independente do sistema operacional (SO) instalado, do tipo de computador ou dispositivo móvel que seja utilizado e da linguagem de programação que tenha sido utilizada na sua criação. O ".NET" oferece a capacidade de desenvolver, implementar, gerenciar e usar soluções conectadas através de Web Services XML, de maneira rápida, barata e segura. Essas soluções permitem uma integração mais ágil entre os negócios e o acesso rápido a informações a qualquer hora, em qualquer lugar e qualquer dispositivo. Todas as aplicações escritas para a ".NET" correm dentro de uma máquina virtual chamada *Common Language Runtime* (CLR). O código que se

encontra a executar aqui dentro chama-se *managed code* e beneficia de várias maneiras como:

1. **Gestão automática de memória:** o CLR dispõe de um "garbage collection" que se encarrega de limpar os objetos que já estão a ser utilizados palas aplicações;
2. **Segurança:** o CLR possui mecanismos que permitem atribuir permissões ao código, baseadas na sua proveniência e em quem o está a executar. Existem também mecanismos que permitem garantir que o código a executar é válido e não irá corromper outros programas que se encontrem a executar no CLR;
3. **Tradução de código intermediário (IL) para código nativo:** ao compilar um programa na plataforma .NET, tipicamente, este é traduzido de uma linguagem de alto nível para uma linguagem intermediária chamada MSIL (*Microsoft Intermediate Language*). O CLR possui um compilador *just-in-time* que se encarrega de traduzir o código intermediário para código nativo do processador, antes de o executar;
4. **Carregamento dinâmico de classe:** o CLR torna possível carregar, em tempo de execução, segmentos de código que antes não estavam presentes na máquina virtual.

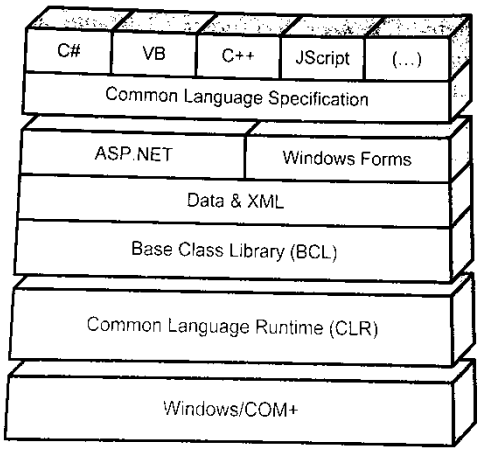


Figura 8 – Arquitetura da plataforma .NET. (Miguel, 2010)

- *Linq* (*Language Integrade Query*) é uma *linguagem integrada de consulta* que permite tratar de forma uniforme dados de diferentes origens (Como exemplo bases de dados ou arquivos XML). Sua ideia principal é resolver um dos maiores problemas de aplicações orientada a objeto que por um lado os objetos são excelentes para manipulação em linguagem de programação, mas por outro lado o armazenamento de dados se faz tipicamente em base de dados relacionais ou arquivos XML, o que aumenta a sua manipulação. O LINQ consiste em uma linguagem declarativa, perfeitamente integrada C#, que minimiza a distância entre o objeto e os dados. (Marques, Pedroso & Figueira, 2009).

2.1.9 Teste

Vários modelos de teste de software foram criados para antever as falhas que podem ser encontradas pelo cliente no momento do seu pleno funcionamento em um ambiente real de trabalho. Teste de carga e estresse, teste de caixa preta e teste de caixa branca são alguns de inúmeros modelos de testes desenvolvidos para garantir a qualidade da aplicação desenvolvida. Com esses modelos de teste, o objetivo da equipe de teste de software é garantir que mesmo nas situações mais extremas e dificilmente alcançadas, aplicação continue executando com a mesmo desempenho de um ambiente ideal.

Os testes realizados em cada fase do ciclo de desenvolvimento de software permitem que um número maior de erros seja descoberto antecipadamente, evitando a migração destes para as fases seguintes. A cultura do teste cria um ambiente favorável para deteção de erros, e identificá-los rapidamente é o objetivo dos profissionais de testes.

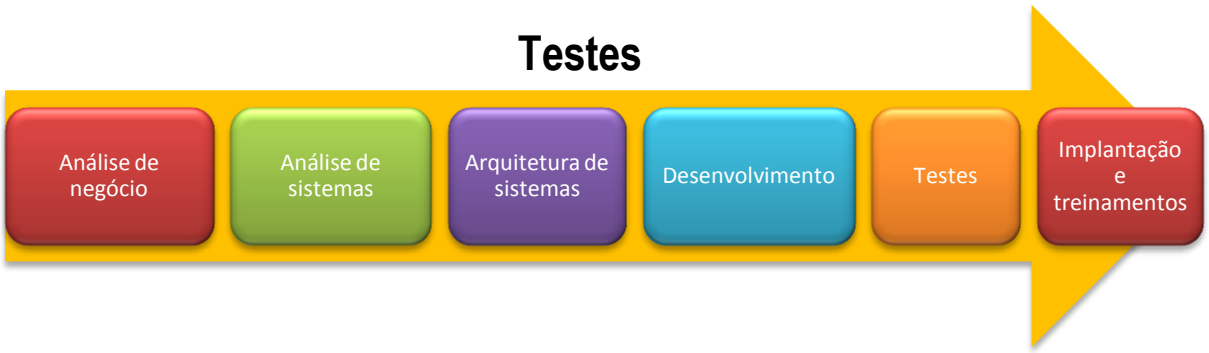


Figura 9 – Rotinas de teste

Os erros ocorrem em todas as fases do processo de desenvolvimento de software. Porém, estudos demonstram que a maior incidência dos erros está concentrada nas fases iniciais do processo de desenvolvimento. Muitos dos erros identificados no produto final são provenientes da má especificação e entendimento sobre os objetivos a serem alcançados.

Se nosso objetivo é reduzir, ao máximo, o nível de erros dentro do processo de desenvolvimento de software, devemos nos concentrar nas atividades iniciais do processo. Desta forma, estaríamos identificando prematuramente os erros impedindo que estes migrassem para outras fases.

Não é possível conceber um processo de teste de software sem integrá-lo com o ciclo de desenvolvimento de software. Um bom processo de teste é aquele que cria uma relação de "um-para-um" entre as fases de desenvolvimento e testes. Esta relação bilateral promove a colaboração entre as áreas e reforça a ideia do objetivo comum.

2.1.10 Implantação

O momento da implantação de um sistema, por incrível que pareça, vai muito além da instalação do sistema na máquina a ser executada. É nesse momento onde os maiores cuidados com as informações contidas nas bases de dados devem ser tratados com maior cautela para não serem danificadas ou até mesmo perdidas.

Além de garantir o pleno funcionamento da aplicação no ambiente solicitado é nessa fase que a integração com todos os sistemas necessários deve funcionar plenamente, principalmente as funcionalidades de segurança e acesso a dados. Isso inclui planos de implantação e até mesmo planos de contingência, ou seja, procedimentos a se realizar caso a aplicação implantada não funcione corretamente ou apresente mais de 50% de erros, o que a torna ineficaz para a utilização do cliente. Em alguns casos de contingência, até o retorno da versão anterior do software é necessária.

Contudo também é na fase de implantação que se contempla o treinamento dos colaboradores para a nova aplicação. De acordo com os processos do ambiente que será implantada a aplicação, o treinamento para as novas funcionalidades desenvolvidas, controle das atividades e novo ambiente de interação com o utilizador deve ser apresentado e esclarecido como também a opção de ajuda de fácil entendimento.

2.2 A Gestão de Projetos

As pessoas às vezes confundem projetos com operações, é claro que ambas as entidades citadas convergem para o alcance de um único objetivo, porém a principal diferença entre elas é que os processos são contínuos e repetitivos, já os projetos são empreendimentos temporários com o objetivo de criar um produto ou serviço único com início e fim definidos (PMI, 2008).

De acordo com Heldman (2005), gerenciamento de projetos é uma metodologia que objetiva de atender os requisitos do projeto para a satisfação do cliente por meio de planeamento, execução, monitoração e controle dos processos utilizando de maneira mais eficiente os recursos disponíveis.

Contudo, consiste em um conjunto de regras que direcionam o acompanhamento dos processos a serem realizados durante o seu ciclo de vida, que consiste na aplicação de conhecimentos, habilidades, ferramentas e técnicas adequadas às atividades do projeto, a fim de atender aos seus requisitos. Sendo essas regras padrões publicadas no Guia PMBOK do Conjunto de Conhecimentos em Gerenciamento de Projetos (PMI, 2008).

O PMBOK (PMI, 2008) formaliza diversos conceitos em gerenciamento de projetos, como a própria definição de projeto e do seu ciclo de vida. Também identifica na comunidade de gerenciamento de projetos um conjunto de conhecimentos amplamente reconhecido como boa prática, aplicáveis à

maioria dos projetos na maior parte do tempo. Estes conhecimentos estão categorizados em nove áreas e os processos relacionados são organizados em cinco grupos ao longo do ciclo de vida do projeto.

Dentre as **áreas de conhecimento** que caracterizam os principais aspetos envolvidos em um projeto e no seu gerenciamento Escopo, Tempo, Custos e Qualidade são os principais determinantes para o objetivo de um projeto: entregar um resultado de acordo com o escopo, no prazo e no custo definidos, com qualidade adequada; em outras palavras, o que, quando, quanto e como. Recursos Humanos e Aquisições são os insumos para produzir o trabalho do projeto. Comunicações e Riscos devem ser continuamente abordados para manter as expectativas e as incertezas sob controlo, assim como o projeto no rumo certo. A Integração abrange a orquestração de todos estes aspetos.

A aplicação dos conhecimentos requer a adoção eficaz de processos apropriados. Cada área de conhecimento abrange diversos processos na gestão de projetos.

De acordo com Miguel (2010) um **processo** é um conjunto de ações e atividades inter-relacionadas que são executadas para alcançar um objetivo. Cada processo é caracterizado por suas entradas, as ferramentas e as técnicas que podem ser aplicadas, e as saídas resultantes.

Os cinco grupos de processos de gerenciamento de projetos, Iniciação, Planeamento, Execução, Monitoramento e Controle e por fim Encerramento têm grande correspondência com o conceito do Ciclo PDCA (Plan - Do - Check - Act): Planejar - Fazer - Verificar - Agir (corrigir e melhorar). O grupo de Planeamento corresponde ao Planejar; Execução, ao Fazer; e Monitoramento e controle englobam Verificar e Agir. E como a natureza dos projetos é finita, o PMBOK ainda caracteriza os grupos de processos que iniciam (Iniciação) e finalizam (Encerramento) um projeto.

2.2.1 A Gestão de Projetos de Produção de Sistemas de Software

A atividade mais importante para o *sucesso da execução* de qualquer projeto é o seu gerenciamento, no entanto, uma gestão mal sucedida na maioria das vezes é responsável pela inviabilidade de um projeto de software a ser desenvolvido. Frame (1994) afirma que projetos falham por duas causas: *Falha de estimação* e *falha na implementação*. Essas falhas se desdobram em um conjunto bem maior:

- Estimativas de prazo e custo não são revisadas após a inserção de novas informações no âmbito do projeto.
- Os gestores de projetos não possuem a formação necessária para direcionar e acompanhar execução do projeto.
- A metodologia de gestão de projetos não é posta em prática de forma correta.
- Mudanças constantes no âmbito do projeto.
- Metodologia de engenharia de software inadequada para a equipe de desenvolvimento.

- Os requisitos não são bem definidos ou sofrem constantes alterações.
- Os testes realizados não atingem um nível de qualidade satisfatório.

Tais falhas bem como seu conjunto de erros agregados contribuem de fato para o insucesso do desenvolvimento do projeto de software. Encontrar e aprimorar soluções para reparar as falhas em cada uma das fases do processo de desenvolvimento (planeamento, levantamento e especificação dos requisitos, implementação, teste e implantação) diminuiria o número de falhas no decorrer do processo de desenvolvimento, porém não se teria o acompanhamento e controle do processo como um todo e correria o risco de cada fase do processo ter características de metodologias de desenvolvimento distintas.

Miguel (2010) afirma que para ser eficaz, a gestão do projeto tem de se conectar em quatro fatores cruciais: pessoas, produto, processo e projeto. Esses fatores correspondem ao que de fato, se bem gerido, é necessário para o sucesso no planeamento, desenvolvimento e implantação de um projeto. Cada fator tem sua importância que interfere diretamente no âmbito do projeto, planeamento, controle, definição dos requisitos, custos, prazos, desenvolvimento e qualidade do produto.

2.2.2 Fatores do projeto

Para a eficácia do projeto de software, cabe ao gestor de projeto gerenciar quatro fatores cruciais de forma que todo o esforço a ser aplicado encaminhe o projeto para a sua construção com sucesso.

- *Pessoas* ou *recursos humanos* para o projeto, três itens devem ser analisados com muita atenção pelo gestor de projetos: utilizadores insatisfeitos, conflitos frequentes entre colaboradores da equipe de desenvolvimento e equipes desmotivadas com baixo nível de produção. Há muito tempo se fala em motivar as pessoas que trabalham com sistemas informáticos. O fator pessoa para o desenvolvimento de uma aplicação é tão importante que existe metodologias voltadas para a melhoria da eficiência organizacional de empresas de software, auxiliando – as encontrar, desenvolver, estimular e reter bons colaboradores para tornar eficiente as suas capacidades de desenvolvimento de sistemas (Curtis, Hefley & Miller, 2001). Cabe então ao gestor de projetos, além de ter competências de acompanhamento, controle e gestão do projeto, a competência de gestão em recursos humanos para a que esses fatores não interfiram no índice de produtividade e maturidade da equipe e principalmente acompanhar se a expectativa do cliente será atendida.
- O *Produto* ou *objeto de desejo* do cliente antes de ser planejado deve-se estabelecer os objetivos e as necessidades que se deseja atingir (definir seu âmbito). Este fator é importante por proporcionar a visibilidade necessária do projeto, consequentemente estimar custos

necessários, avaliar os riscos, verificar de fato as atividades necessárias para realização e por fim se ter o cronograma estimativo com a data a ser entregue o produto. Esta fase se inicia com o engenheiro de sistemas definindo com o cliente/utilizador quais as necessidades reais que o projeto deve atender. Com estas informações, para além de se especificar o objetivo principal do projeto, também se definem os limites do mesmo, de acordo com os requisitos apontados pelo utilizador.

- O *Processo de desenvolvimento de software* assim como em qualquer processo produtivo (seja automobilístico, têxtil ou manufaturas em geral) independente da metodologia a ser utilizada, consiste de atividades que são comuns e devem ser realizadas no decorrer do processo de desenvolvimento do sistema. Conhecer o processo de produção definindo a estrutura do mesmo independente da complexidade, dimensão do software e a metodologia utilizada, acrescenta maior fiabilidade no sistema informático, pois são nos processos que são definidos níveis de qualidade que se desejam atingir durante sua execução.
- O *Projeto* é o artefacto que engloba todos os fatores descritos anteriormente o projeto reúne todas as metodologias a serem executadas no decorrer do desenvolvimento do sistema, porém essas práticas utilizadas não garantem a eficiência do processo de gestão do projeto. The Standish Group (2008) afirma que 20% dos projetos de software fracassam completamente e que 44% enfrentam grandes problemas de custo e prazo no seu processo de desenvolvimento. Mesmo que essas taxas tenham melhorado através da aplicação de novas metodologias, a escassez de sistemas de especificação de projetos de software ainda é muito grande comparado a projetos de engenharia de outros segmentos de produção. Keil (1995).

Para que as falhas sejam evitadas, cabe ao gestor de projeto desenvolver juntamente com a equipe práticas preventivas de riscos nos projetos e tomar conhecimento de fatores que podem vir a influenciar no processo de desenvolvimento de sistemas, como segue abaixo:

- Estabelecer um canal de fácil comunicação entre a equipe;
- Evitar acúmulo de sinais de riscos;
- Compreender fatores críticos que contribuem para a uma boa gestão de projetos.
- Desenvolver metodologias coerentes para o planeamento, controle e implantação do projeto.

No decorrer deste trabalho, analisaremos os resultados da implantação de um sistema que realiza o planeamento do desenvolvimento de um projeto de software bem como o acompanhamento das atividades desenvolvidas (planeamento e engenharia de software, o levantamento e a especificação dos requisitos, o desenvolvimento de uma aplicação, o teste de qualidade de software e a implantação

e manutenção do sistema) no decorrer de sua construção, realizada no ambiente de uma empresa de desenvolvimento de sistemas.

2.2.3 Definição do âmbito do projeto

A definição do âmbito do projeto ou especificação das necessidades do cliente é realizada com o objetivo de recolher informações necessárias para o desenvolvimento do produto de software. É nesse momento que o gestor de projeto de software juntamente com o analista de negócio deverão empregar as técnicas mais adequadas para garantir que os requisitos recolhidos correspondem as necessidades e expectativas dos stakeholders (clientes), e são precisos e completos.

Diversas técnicas são aplicadas para este processo:

- Entrevistas;
- Reuniões de *brainstorming*;
- Questionários e inquéritos;
- Elaboração de protótipos;
- Entre outros.

A documentação final após estes processos devem descrever como os requisitos especificados satisfazem o processo de negócio que o cliente solicitou. Geralmente se inicia com o nível baixo de detalhe e torna-se cada vez mais detalhado a medida que se conhece mais o projeto. Os requisitos devem cumprir um conjunto importante de critérios, antes de serem submetidos a desenvolvimento.

- Mensuráveis;
- Auditáveis;
- Completos;
- Consistentes;
- Aceitáveis;

O documento que define o âmbito do projeto de software a ser desenvolvido por este projeto é apresentado no item *"*Análise de Negócio e Modelação de Processos" que define as atribuições que contemplam esta fase do processo de desenvolvimento de software.

A descrição do âmbito do projeto é um documento dinâmico, que começa a ser elaborado na fase inicial do projeto e vai sendo modificado à medida que os detalhes do projeto são conhecidos. Apenas no final da fase de planeamento, quando se conhecem todas as condições e limitações do projeto é possível ter um documento completo.

Os objetivos principais da descrição do âmbito do projeto é comunicar a todos os interessados:

- Os objetivos do projeto;
- O que ele deve produzir e entregar;

- Os utilizadores envolvidos;
- Os prazos e os custos;
- E por fim como o projeto será gerenciado.

O nível de detalhe do projeto a ser desenvolvido define o grau de prioridade para início de construção, fornecendo ao gestor do projeto bases para controlar o âmbito geral do trabalho. Este procedimento é importante, pois determina a eficiência da equipe de desenvolvimento para o planeamento do projeto.

2.2.4 Estimação de prazos, recursos e custos

A estimação eficaz do esforço (recursos e prazos) e custo do software constitui uma das atividades simultaneamente mais difíceis e mais importantes do desenvolvimento de sistemas de software. (Miguel, 2010).

O desafio maior em estimar os indicadores de produção de software são compreender o domínio do software (suas regras de negócio) e a capacidade da equipe em fornecer soluções de software para as necessidades especificadas pelo cliente em um determinado ambiente. Só então é possível prever a quantidade de esforço necessário para o desenvolvimento do produto.

Porém, em qualquer área de desenvolvimento (seja civil, industrial, elétrico, sistemas, entre outras) o processo de estimação se baseia num conjunto de técnicas e procedimentos que a organização se baseia para enfim chegar ao valor estimado. A estimativa não é e nunca será uma ciência exata, em virtude de existirem demasiadas variáveis a interferir diretamente no esforço necessário para o desenvolvimento do projeto de software e consequentemente os seus respetivos custos – técnicas de desenvolvimento, políticas organizacionais, políticas e humanas.

De acordo com a definição de estimar – a melhor visão do futuro, baseada no conhecimento e informações disponíveis hoje – não se espera que se acerte sempre, o que se espera é acertar de uma maneira geral.

Podemos listar diversas dificuldades encontradas para estimar os esforços e custos do projeto de software, entre elas estão:

- Para se realizar corretamente, a estimativa exige um volume significativo de esforço;
- A estimação é frequentemente feita de uma forma apressada, sem uma adequada avaliação dos esforços exigidos;
- Para se desenvolver estimativas é necessário possuir experiências anteriores, em especial para grandes projetos;
- O ser humano está sujeito a enviesamento, ou seja, quem estima tende a considerar a duração estimada de uma certa porção do trabalho e depois extrapola simplesmente esta estimativa para o restante do sistema, ignorando os aspetos lineares do desenvolvimento sistema.

No entanto, a estimativa do projeto de software pode ser transformada em uma numa série de passos sistemáticos que forneçam níveis aceitáveis de riscos. Para isso algumas regras simples estão dispostas:

- Basear-se nas estimativas de projetos concluídos anteriormente;
- Usar técnicas de decomposição relativamente simples;
- Usar um ou mais modelos de estimação;

Uma prática em crescente utilização para estimativa do esforço e dimensionamento das funcionalidades desejadas em um sistema é a análise de ponto de função.

2.2.5 Análise de ponto de função

Ainda na fase de análise de negócio, após o entendimento de todas as necessidades do cliente, é necessário se medir o tamanho da aplicação a ser construída. Para isso alguns métodos são disponibilizados ao analista. Uma desses métodos é a estimativa por experiência, onde uma equipe de desenvolvimento que já construiu funcionalidades semelhantes às solicitadas atribui o tempo necessário para a construção da aplicação. No entanto, algumas funcionalidades podem ficar superestimadas, comprometendo custo e prazo de entrega.

Outra opção é a utilização da análise de ponto de função (APF) que mede o tamanho das funcionalidades solicitadas pelo ponto de vista dos utilizadores que estão propostas nos modelos de layouts aprovados junto ao cliente. Suas dimensões são calculadas junto a uma tabela de complexidade onde a mesma disponibiliza a estimativa do esforço necessário para desenvolver a funcionalidade desejada.

A APF foi à conclusão de estudos realizados na década de 70 por Allan Albrecht, da IBM, diante da necessidade de se ter uma abordagem para quantificar o esforço de desenvolvimento de funcionalidade de sistema computacional independente da linguagem que se está utilizando (IFPUG, 2004).

Ponto de função é a medida do tamanho da aplicação computacional, que por sua vez, é medido de um ponto de vista funcional do utilizador. É independente de qualquer característica de dessenvolvimento (da linguagem computacional, da metodologia de desenvolvimento, da tecnologia ou da capacidade do grupo de desenvolvimento de desenvolver a aplicação). O fato do ponto de função ser usado originalmente para predizer esforço, isso nada mais é que uma consequência do fato de que tamanho é geralmente um indicador associado ao esforço do desenvolvimento.

É importante salientar o que os pontos de função não são medidas perfeitas de esforço para desenvolver uma aplicação ou seu valor de negócio, uma vez que por mais que o sistema tenha suas funcionalidades definidas, a sua construção depende de desenvolvedor para desenvolvedor (IFPUG,

2004). Isso é facilmente ilustrado com uma analogia às medidas em construções, onde em uma casa de cem metros quadrados quase sempre é mais barata do que uma casa de duzentos metros quadrados. Porém, muitos requisitos como os materiais utilizados podem fazer a casa menor mais cara. Outros fatores como número de casas de banho e localização podem fazer a casa de menor dimensão ter um custo mais elevado. Abaixo seguem algumas razões pelas quais a abordagem de APF deve ser utilizada para a estimativa de desenvolvimento de sistemas de software:

- Medir produtividade.
- Estimativa de desenvolvimento e suporte.
- Monitorizar alterações de origem externa.
- Normalização das formas de medida.

Sua contagem se inicia primeiramente classificando os componentes presentes na proposta de interface homologada. Após a identificação dos componentes, o mesmo é multiplicado pela sua faixa de esforço a realizar e por fim, multiplicado pelo custo por ponto de função para calculo de custo e/ou pelo tempo por ponto de função para calculo de prazo de entrega. Os pontos de funções classificam os componentes nas seguintes funcionalidades:

- Entrada Externa (EE) – é um processo em que os dados atravessam a camada de fronteira da aplicação de fora para dentro. Tais dados podem vir de uma tela de entrada de dados, por via eletrônica ou através de outro aplicativo podendo ser informações de controle ou informações do negócio. No caso dos dados serem informações do negócio, será utilizado para armazenar um ou mais arquivos (ALI). Se os dados forem informações de controle, não será necessário que armazenem arquivos.
- Saídas Externas (SE) – é um processo em que os dados passam através da camada de fronteira, de dentro para fora traduzidos em relatórios ou arquivos de saída, que são enviados a outros aplicativos. Esses relatórios ou arquivos são originados a partir de um ou mais arquivos lógicos internos e/ou arquivos de interface externa.

  Por sua vez, dados derivados são informações cujo processamento vai além da recuperação e edição direta de registros (consulta e/ou cadastro) de arquivos lógicos internos ou arquivos de interface externa. É o resultado de um eventual processamento ou execução de funcionalidade de negócio, que ocorrem quando um ou mais elementos são combinados com uma fórmula, de modo a gerar ou derivar um ou mais elementos de dados adicionais.
- Consulta Externa (CE) – é um processo entrada e saída, que resulta na recuperação de dados (consulta) de um ou mais arquivos lógicos internos e/ou arquivos de interface externa que por sua vez é enviada para fora da camada de fronteira da aplicação.

- Arquivo Lógico Interno (ALI) – um conjunto de dados relacionados (entidades), identificável pelo utilizador, que reside inteiramente dentro da camada de banco de dados da aplicação é mantido através de Entradas Externas.
- Arquivo de Interface Externa (AIE) – um conjunto de dados relacionados (entidades), identificável pelo utilizador, que é utilizado apenas para referência, pois residem inteiramente fora da aplicação e são mantidos por outras aplicações. O AIE é um Arquivo Lógico Interno para outra aplicação.

A Figura 10 apresenta uma visão geral dos tipos de função que são considerados na análise de ponto de função.

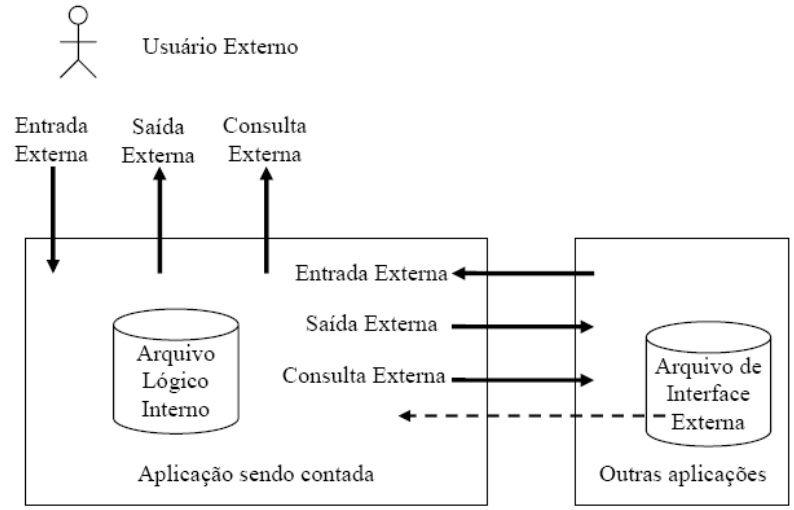


Figura 10 – Visão geral das funcionalidades de uma aplicação segundo a análise APF (Dias, 2012).

Na Figura 11 podemos observar o fluxo dos passos necessários para a realização da contagem dos pontos de função da aplicação.

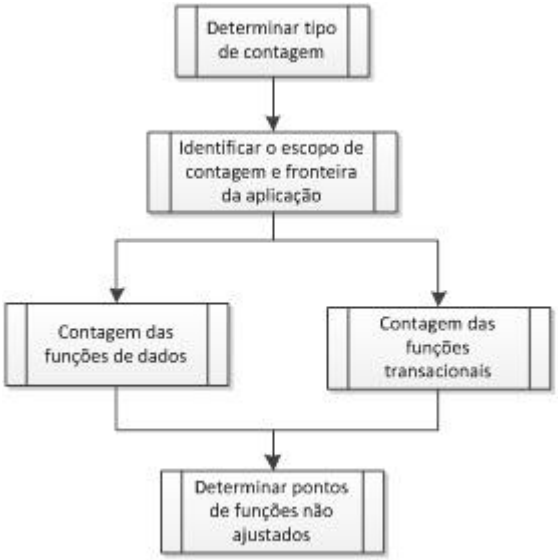


Figura 11 – O Procedimento de Contagem de Pontos de Função, adaptado (Dias, 2012).

O processo de contagem de pontos de função de acordo com (Dias, 2012) se dá através de dois passos:

1. O primeiro consiste em identificar arquivos lógicos internos (ALIs) e arquivos de interface externa (AIEs) que deve ser classificada em dois conceitos: registros lógicos e itens de dados.
    - Registros Lógicos são subconjuntos de dados dentro de um ALI/AIE, que foram reconhecidos pelo utilizador.
    - Um Item de Dados, por sua vez, é um campo reconhecido pelo utilizador como único e não repetido. Vale destacar que só devem ser contados os itens de dados utilizados pela aplicação em contagem.

Uma vez identificando-se os registros lógicos e os itens de dados de um ALI/AIE, pode-se verificar sua complexidade relacionando a quantidade de registros com a quandidade de dados manipulados, utilizando como referência a classificação apresentada na Tabela 3.

Tabela 3 - Identificação da Complexidade das Funções de Dados (Dias, 2012).

Número de registros lógicos	**Número de itens de dados**		
	De 1 a 19	**De 20 a 50**	**51 ou mais**
Apenas 1	Simples	Simples	Média
De 2 a 5	Simples	Média	Complexa
6 ou mais	Média	Complexa	Complexa

2. O segundo passo envolve a contagem de funções que atravessam as camadas do sistema (entradas externas, saídas externas e consultas externas) e sua classificação se dá de acordo com a complexidade funcional (simples, média ou complexa) baseada no número de arquivos referenciados e dos itens de dados manipulados pela função, utilizando a Tabela 4 para entradas externas e a Tabela 5 para saídas e consultas externas.

Nessas tabelas um arquivo referenciado pode ser um ALI/AIE lido ou um ALI mantido pela função. Já o número de dados é considerado apenas os dados efetivamente referenciados pelas funções nos arquivos em quetão.

Tabela 4 - Identificação da Complexidade de Entradas Externas (Dias, 2012).

Número de arquivos referenciados	**Número de itens de dados referenciados**		
	De 1 a 4	**De 5 a 15**	**16 ou mais**
0 ou 1	Simples	Simples	Média
2	Simples	Média	Complexa
3 ou mais	Média	Complexa	Complexa

Uma vez quantificadas às funções, é possível calcular os PFs de uma aplicação. Esse cálculo é feito a partir dos totais de pontos de função (NPFi) e do total de pontos de função não ajustados.

Para cada um dos cinco tipos de função (ALI, AIE, EE, SE e CE), são computados os totais de pontos de função (NPFi), segundo a seguinte expressão de contagem de acordo (Dias, 2012):

$$NPF_i = \sum_{j=1}^{3} NC_{i,j} * C_{i,j}$$

onde NCi,j = número funções do tipo i (i variando de 1 a 5, segundo os tipos de função existentes: ALI, AIE, EE, SE e CE) que foram classificados na complexidade j (j variando de 1 a 3, segundo os valores de complexidade: simples, média e complexa).

Ci,j = valor da contribuição da complexidade j no cálculo dos pontos da função i, dado pela Tabela 5.

Tabela 5 - Identificação da Complexidade de Saídas e Consultas Externas (Dias, 2012).

Número de arquivos referenciados	**Número de itens de dados referenciados**		
	De 1 a 5	**De 6 a 19**	**20 ou mais**
0 ou 1	Simples	Simples	Média
2 ou 3	Simples	Média	Complexa
4 ou mais	Média	Complexa	Complexa

O total de pontos de função não ajustados (PFNA) é dado pelo somatório dos pontos das tabelas de função de contagem de acordo (Dias, 2012):

$$PFNA = \sum_{i=1}^{5} NPF_i$$

sendo que i varia de 1 a 5, segundo os tipos de função existentes (AIL, AIE, EE, SE e CE).
Por não influenciar em mais de 10% as métricas de estimativas, a contagem de pontos de função não ajustados não é uma abordagem muito utilizada atualmente.

Tabela 6 - Contribuição das Funções na Contagem de PF's Não Ajustados (Dias, 2012).

Função	**Complexidade**		
	Simples	**Média**	**Complexa**
ALI	7	10	15
AIE	5	7	10
EE	3	4	6
SE	4	5	7
CE	3	4	6

2.2.6 Planeamento do projeto (cronograma)

Após definir o âmbito do projeto, funcionalidades, estimar prazos, recursos e tempo, chega a hora de planejar as atividades necessárias para o desenvolvimento do sistema. Seu objetivo é determinar a ordem em que as atividades inerentes ao sistema sejam executadas. Há duas formas de apresentar o cronograma de execução do projeto:

- Gráfico de Gantt
- Diagrama de rede

O *gráfico de Gantt* é eficaz para projetos simples e de curta duração. É atribuída, pelo gestor, uma barra retangular para cada atividade que em seguida são dispostas horizontalmente ao longo de uma linha do tempo na ordem em que as mesmas devem ser executadas, podendo algumas atividades ser colocadas na linha do tempo de forma concorrente com outras. A sequência é muitas vezes ditada pela disponibilidade dos recursos que por qualquer outra consideração. (Miguel, 2010).

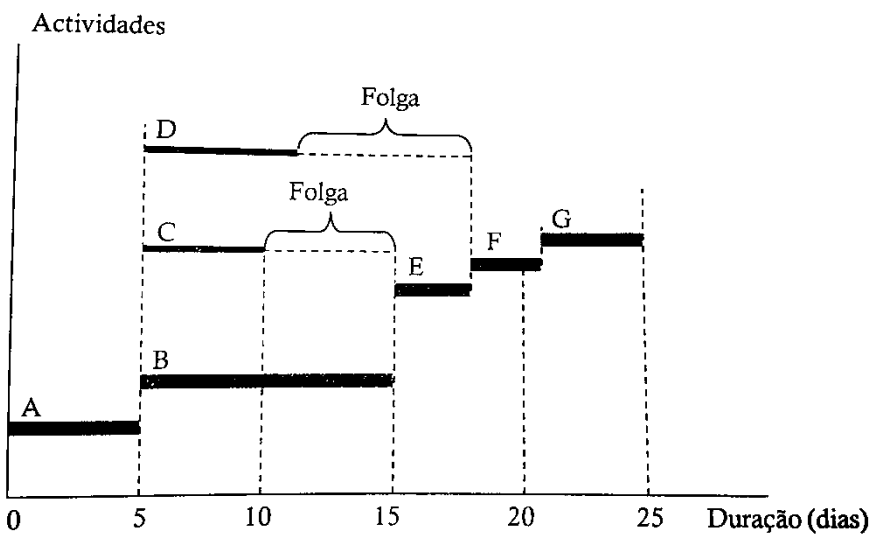


Figura 12 – Gráfico de Gantt das atividades de um projeto (Miguel, 2010).

No entanto, o gráfico de Gantt apresenta duas limitações:

- Primeiro que pela sua simplicidade não apresenta informações detalhadas da atividade, apenas reflete a ordem das atividades a serem desenvolvidas.
- Segundo, o gráfico de Gantt não informa se o cronograma de que resulta o gráfico conclui o projeto no prazo mais curto possível, nem se os recursos são utilizados de forma mais eficaz.

Embora o gráfico de Gantt seja de simples de criar e não exija o uso de ferramentas informatizadas, recomenda-se o uso de diagramas de redes, pois:

- Possibilita uma imagem da sequência em que flui o trabalho do projeto.
- Inclui informações detalhadas.
- Serve como ferramenta de analítica para calendarização do projeto e para problemas de gestão de recursos, à medida que surgem durante a vida do projeto.
- Possibilita o cálculo do prazo mais curto para a finalização do projeto.

As atividades com suas respetivas durações são de fato informações de suma importância para o gestor de projetos de software principalmente para a construção da sua imagem gráfica, que proporciona a visão de duas informações adicionais a cerca do projeto:

- O primeiro momento em que pode ser iniciado o trabalho em cada atividade.
- A primeira data de conclusão que é expectável para o projeto.

Um *diagrama de rede do projeto* é uma representação figurativa da sequência em que podem ser executadas as várias atividades que compõe o projeto. Há um conjunto reduzido de regras simples que é necessário seguir para construir esse diagrama, dentre elas a precedência das atividades, o tempo

de cada uma e uma análise de caminho crítico para acompanhar o tempo mais curto e mais longo de realização das atividades.

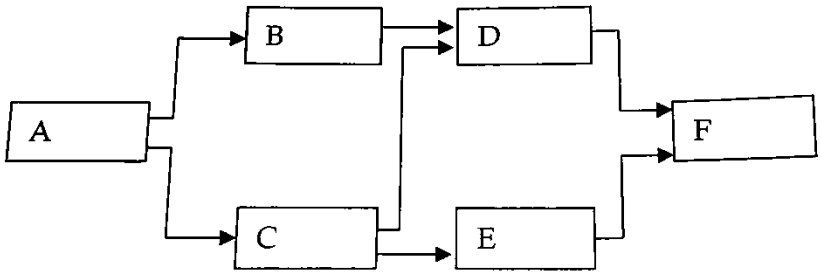


Figura 13 – Diagrama de rede de projeto. (Miguel, 2010)

Para este projeto, utilizaremos o gráfico de Gantt, porém iremos disponibilizar no mesmo as informações necessárias de acompanhamento das atividades por parte do gestor. Prazos e recursos são definidos pela quantidade de ponto de função definida pela métrica do tamanho do software.

2.2.7 Acompanhamento, controle e encerramento do projeto

Até este ponto, a descrição do esforço despendido na gestão do projeto de software está relacionado com a análise de listagem dos requisitos (na gestão do âmbito do projeto), estimação dos recursos, prazos, custos do projeto e o planeamento das atividades a serem realizadas no prazo estabelecido para o desenvolvimento do sistema.

Cabe agora ao gestor do projeto garantir que todas as atividades planeadas sejam desenvolvidas sem impedimentos através de acompanhamento contínuo, controlando os possíveis impedimentos e desenvolvimento das atividades até a finalização do projeto (desde a aceitação do cliente até a implantação do sistema). Com o apoio do gráfico de Gantt, este trabalho apresenta em seu desenvolvimento prático uma ferramenta que dispõe de um sistema de software direcionado à gestão de projetos de software que lista todas as atividades referentes ao projeto e principalmente o estado de cada uma dela, consequentemente a comparação entre o nível de desenvolvimento previsto e o nível de desenvolvimento real.

Essa ferramenta de controle de atividades de projeto garante ao gestor visibilidade das atividades inerentes ao projeto, o acompanhamento das atividades realizadas, em execução e a serem executadas, possibilitando verificação de fases de processo que precisam de adequações objetivando o desenvolvimento pleno da produção de software. Com a identificação das atividades que comprometem o prazo de desenvolvimento sistema, torna mais visível ao gestor de projeto realizar ações que contornem as situações que comprometem os indicadores já definidos para a construção do

sistema. É nessa fase que se aplica a gestão de risco no desenvolvimento do projeto. Ao gestor cabe buscar soluções aos problemas encontrados para que nos próximos projetos a serem desenvolvidos, mesmo encontrando as mesmas dificuldades, tal problema já apresente uma proposta de solução.

A partir do acompanhamento e controle de sistema, o seu encerramento pode ser planejado, desenvolvendo workshops de apresentações do software construído já identificando as limitações encontradas nas fases de desenvolvimento do sistema, o que posteriormente facilitaria o desenvolvimento de novas atualizações e principalmente nas novas necessidades de negócio do cliente para o sistema de software.

2.2.8 Gestão de equipas

A equipa de desenvolvimento de projetos de software compreende todos os recursos disponíveis para o gestor. Claro, o ambiente de desenvolvimento da empresa é importante, a arquitetura de desenvolvimento, a linguagem a ser utilizada, os modelos de documentação e registro dos requisitos do cliente, as técnicas de teste de software, são componentes importantíssimos para o desenvolvimento de sistemas de software, mas que só serão eficientes se realizados por equipas eficientes.

No entanto, gerir equipas é a parte da gestão de projetos de sistemas que necessita de maior atenção do gestor, pois duas situações podem frequentemente acontecer em ambientes de desenvolvimento de sistemas:

- Conflitos frequentes entre colaboradores da equipe de desenvolvimento;
- Equipes desmotivadas com baixo nível de produção;

É comum em empresas de desenvolvimento de software haver competições entre equipas para entrega de seus produtos antes da outra, o que por um lado é saudável, pois aumenta o nível de produção, mas por outro faz com que os conflitos, originados geralmente por falhas executadas por um colaborador em uma das fases de processo de desenvolvimento, afetem o ambiente de desenvolvimento ou até mesmo o ambiente de toda a organização. Solucionar essa situação logo no início é uma função muito importante do gestor e de preferência com uma solução que objetive o bem-estar tanto do projeto quando da equipe.

Porém, mais difícil é quando se tem equipes desmotivadas. É sinal que o nível de produção do desenvolvimento de sistemas já diminuiu significativamente. Solucionar esse empasse já não é simplesmente um desentendimento. Dependendo do âmbito da desmotivação (um membro da equipe ou toda a equipe) as soluções podem ir de uma simples meta até as mudanças nas fases dos processos de desenvolvimento.

Estabelecer soluções prontas (dependendo de cada organização) já esperando essas situações é a melhor alternativa que o gestor de projetos pode encontrar para resolver tais problemas. No entanto, muita atenção nos recursos e acompanhar continuamente o processo de desenvolvimento consistem na melhor alternativa para contornar os possíveis conflitos.

3 METODOLOGIA

3.1 Contexto do Projeto de Investigação

Manaus é um município brasileiro, capital do estado do Amazonas e o principal centro financeiro, corporativo e econômico da Região norte do Brasil. É uma cidade histórica e portuária, localizada no centro da maior floresta tropical do mundo. Situa-se na confluência dos rios Negro e Solimões. É a cidade mais populosa da Amazônia, de acordo com o IBGE, sendo uma das cidades brasileiras mais conhecidas mundialmente, principalmente pelo seu potencial turístico e pelo ecoturismo, sendo o décimo maior destino de turistas no Brasil. Está localizada no extremo norte do país, a 3.490 quilômetros da capital federal, Brasília.

Com a implantação da Zona Franca de Manaus na década de 1960, a cidade ocupou lugar de destaque entre as mais ricas do Brasil e da América Latina. Ao lado de Cuiabá, capital de Mato Grosso, é a capital que mais cresceu economicamente nos últimos quarenta anos, fato explicado principalmente pela implantação e desenvolvimento da Zona Franca de Manaus, que também atraiu milhares de migrantes que ocuparam de forma desordenada a periferia da cidade.

A Zona Franca de Manaus (ZFM), também conhecida como Polo Industrial de Manaus, é um centro financeiro (o principal da região norte do Brasil) implantado pelo governo brasileiro objetivando viabilizar uma base econômica na Amazônia Ocidental, promover a melhor integração produtiva e social dessa região ao país, garantindo a soberania nacional sobre suas fronteiras. É um dos mais modernos da América Latina. A mais bem-sucedida estratégia de desenvolvimento regional, o modelo leva à região de sua abrangência (estados da Amazônia Ocidental: Acre, Amazonas, Rondônia e Roraima e as cidades de Macapá e Santana, no Amapá) desenvolvimento econômico aliado à proteção ambiental, proporcionando melhor qualidade de vida às suas populações.

Com esse cenário, muitas empresas que se instalaram no polo industrial de Manaus necessitaram de sistemas que potencializassem e apoiassem seus negócios de maneira mais eficiente o que necessitou e abriu mercado para empresas de desenvolvimento de sistemas, que por sua vez diante da grande demanda não conseguem atender às necessidades de mercado. Muitas dessas dificuldades relacionadas à falta de suporte aos seus processos de produção.

Este projeto de investigação compreendeu uma empresa de desenvolvimento de sistemas localizada na cidade de Manaus composta por: um gerente de desenvolvimento de sistemas, dois arquitetos de software, um analista de sistemas, um designer gráfico, um desenvolvedor e um analista de teste totalizando uma equipe com sete membros e um projeto de sistema de software em produção.

3.2 Caracterização da pesquisa

Uma pesquisa se desenvolve diante a associação de conhecimentos disponíveis (sejam bibliográficos entre outros) em conjunto com metodologias, métodos, técnicas e entre outros recursos científicos. No entanto a pesquisa só se torna satisfatória quando a mesma atende a cada uma das fases de seu planeamento, desde a identificação e formulação do seu problema até a apresentação dos resultados obtidos.

Por sua vez, as pesquisas são classificadas e distribuídas em três grandes grupos: descritivas, explicativas e exploratórias. Nesse caso, apresenta-se um estudo exploratório que, conforme Seltiz et al. (1967, p. 63):

> "Têm o objetivo de proporcionar maior familiaridade com o problema, tendo em vista torna-lo mais explícito ou a construir hipóteses. Pode-se dizer que estas pesquisas têm como objetivo principal o aprimoramento de ideias ou a descoberta de intuições. Na maioria dos casos, essas pesquisas envolvem: levantamento bibliográfico e/ou documental; entrevistas com pessoas que tiveram experiências práticas com o problema pesquisado; e análise de exemplos que estimulem a compreensão".

Em linhas gerais, as pesquisas com esse caráter visam desvendar o que está acontecendo, de forma a questionar o que já está escrito, buscando um aperfeiçoamento de ideias através de uma nova ótica e propondo uma nova ferramenta que auxilie no apoio a tomada de decisão do problema apresentado.

As diversas abordagens representam visões teóricas da realidade e defendem posturas um tanto diferenciadas.

Portanto, quanto a sua natureza essa pesquisa é qualitativa, pois na organização dos dados coletados sobre a gestão de projetos de software, trabalhar-se-á com dados qualitativos e quantitativos uma vez que, com ambas as informações se procura compreender e analisar o foco da pesquisa a partir da realidade de projetos de software desenvolvido utilizando o modelo de software para o acompanhamento das atividades pois esses tipos de pesquisa não são opostos, mas sim complementares "pois a realidade abrangida por eles interage dinamicamente, excluindo qualquer dicotomia" (Minayo, 1994, p.22).

Enquanto instrumentos metodológicos, a pesquisa tomará como ponto de partida a pesquisa bibliográfica, depois a análise dos artefactos documentais referentes à arquitetura do projeto de software e por fim a análise dos dados resultantes referentes ao desenvolvimento de software por uma determinada empresa.

A pesquisa bibliográfica estará voltada para a compreensão do contexto das necessidades da gestão de projetos de software em suas respetivas fases de construção destacando as influências no campo

desenvolvimento. Visa também relacionar tendências de gestão de projetos de software propostos pelos mercados de desenvolvimento de software e os mais usuais modelos de desenvolvimento de sistemas.

O desenvolvimento dos artefactos documentais referentes à arquitetura do projeto de software será realizado junto a referências bibliográfica técnicas direcionadas a área de gestão de projetos de software a fim de levantar informações sobre os processos gestão mais utilizados e eficientes neste cenário. A análise desses artefactos é de extrema relevância nesta pesquisa, pois possibilitam a consulta a informações técnicas, normas, regulamentos, artigos científicos tecnológicos e outros. "Documentos são fontes de informações que ainda não receberam organização, tratamento analítico e publicação. A pesquisa documental é a que se serve dessas fontes". (Santos, 1999, p. 29.).

A análise dos dados referentes ao desenvolvimento de software permite observar o levantamento das informações sobre as necessidades de melhorias nas fases de gestão de projeto de software, levantar requisitos estabelecendo documentos que servirão como base para a construção do sistema, o tempo previsto de desenvolvimento de acordo com os recursos atuais disponíveis, bem como os testes específicos alinhados às necessidades do cliente e por fim o tempo real de desenvolvimento com seus devidos custos.

O artefacto documental inicial do projeto de software definido como *documento de visão* será estabelecido pelos membros da equipe de desenvolvimento observada – 1 gestor de desenvolvimento de sistemas, 1 analista de sistemas, 1 designer gráfico, 2 arquitetos de software, 2 desenvolvedores de sistemas e 1 analista de teste. O uso deste instrumento tem como objetivo levantar os requisitos necessários que possibilite compreender aspetos das diferentes fases do processo de construção do sistema de software que caracterizem o perfil do modelo de gestão de projeto de software da organização como um todo desde o levantamento dos requisitos com os clientes, a modelagem e documentação do sistema, sua arquitetura funcional, seu desenvolvimento, seus teste com níveis de qualidade e treinamentos e implantações.

O Documento de Visão possibilita uma visão geral do objeto pesquisado e permite um maior entendimento dos dados levantados. O referido instrumento, segundo Richardson, (1999, p.191) tem as seguintes características:

> (...) é um dos artefactos da Análise Estruturada para projetos de sistemas informáticos. Ele facilita uma análise preambular deste, sendo de grande relevância durante as primeiras fases, permitindo a captura de todas as perspetivas que o sistema pode abranger. Pretende-se que sirva como ferramenta de auxílio, a evitar alguns dos problemas mais custosos com que as pessoas envolvidas no projeto poderão ter que se confrontar. Esta ajuda é proporcionada através da divulgação do conteúdo deste a todos aqueles que estejam integrados no sistema.

No final dos processos de desenvolvimento do sistema a análise dos dados será direcionada ao modelo de gestão de projeto de software de uma determinada empresa da cidade de Manaus – AM mostrando sua realidade, analisando seu andamento e mostrando o impacto de possiveis mudanças que possam ocorrer durante a sua execução. A ênfase do trabalho final será desenvolver um módulo de software que auxilie o processo de produção de sistemas informáticos mostrando o andamento de suas atividades, custos e prazos apoiando a tomada de decisão para eventuais mudanças no projeto por parte do gestor.

4 ANÁLISE, IMPLEMENTAÇÃO E RESULTADOS DO PROJETO DE INVESTIGAÇÃO

4.1 Análise do problema

Um dos principais problemas da gestão de projetos, além da definição do escopo do projeto, está na gestão das atividades inerentes ao projeto e principalmente o acompanhamento da execução das mesmas. Tal informação é de grande importância para: medir o nível de execução total do projeto, acompanhar os custos aplicados até o momento, comparar o andamento do projeto com os prazos acordados no início do mesmo, acompanhar os níveis de produtividade entre outros índices que a partir do acompanhamento das atividades podem ser medidos.

No desenvolvimento de projetos de software, a gestão das atividades realizadas é importante para os gestores de projeto, pois são através destes acompanhamentos, desde a fase de concepção até a fase de testes, que o gestor pode apresentar resultados físicos sobre o andamento do projeto junto aos seus clientes, utilizadores e interessados.

Inúmeras metodologias de gestão surgiram com a ideia de propor um acompanhamento mais simples e intuitivo das atividades referentes ao projeto, mesmo que os processos da empresa possuíssem processos bem definidos, esses processos eram quebrados em atividades para o acompanhamento. Contudo, por mais que as metodologias tivessem um propósito bem definido o acompanhamento do projeto poderia não funcionar da forma esperada por dificuldade das empresas na adaptação a uma nova cultura que a metodologia exigia.

A Figura 14 apresenta o planeamento das atividades realizadas inicialmente pela equipa de desenvolvimento da empresa que foi objeto deste estudo utilizando os passos da metodologia Scrum para estimar esforço para a realização das atividades e o desenvolvimento de um módulo de sistema. Inicialmente as atividades são especificadas e realiza-se a atribuição do seu respetivo tempo de desenvolvimento através das experiências de implementação de cada colaborador responsável.

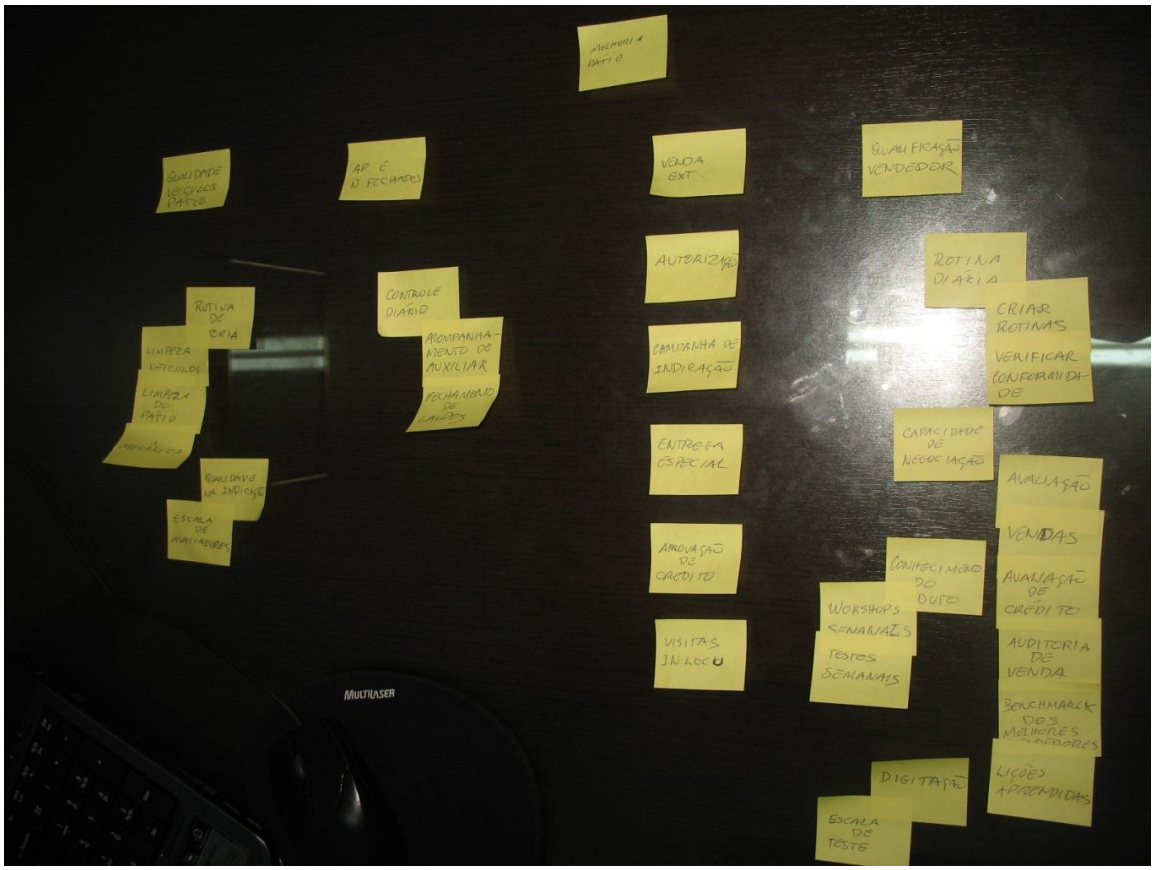


Figura 14 – Quadro de planeamento inicial do projeto

Após a definição das atividades, as mesmas são distribuídas no quadro de acompanhamento de atividades, divididas em estórias do usuário, tarefas a fazer, tarefas em andamento e tarefas finalizadas. As estórias definem o que o usuário de fato deseja em ordem de prioridade, já as tarefas se referem às estórias que refletem o andamento da mesma. Essas por sua vez são divididas em sprints (prioridades de desenvolvimento dentro de um ciclo) que deve ser entregue no final de um determinado período de tempo.

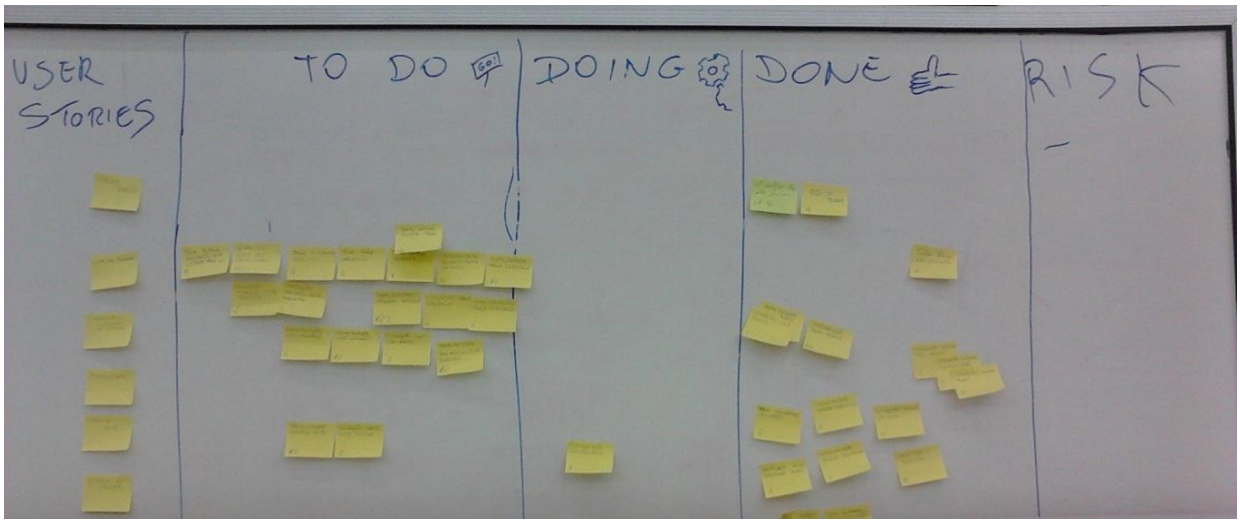


Figura 15 – Quadro de organização do ciclo do projeto

Porém ao longo do desenvolvimento do projeto fatores como a falta de cultura da empresa em atualizar o quadro de atividades, mudança nos requisitos, alteração de prioridades das atividades a serem desenvolvidas influenciam em muito o acompanhamento das atividades do projeto. A Figura 16 demonstra um exemplo do resultado de como excesso de mudanças em um projeto pode impactar no acompanhamento das atividades realizadas.

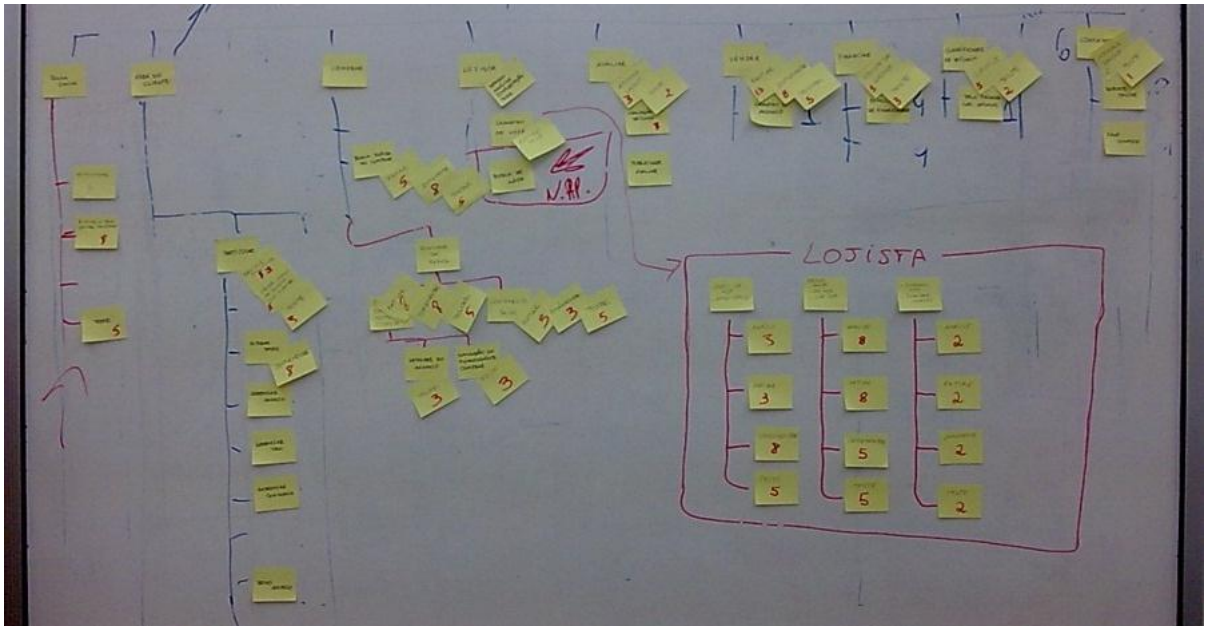


Figura 16 – Quadro de acompanhamento do ciclo do projeto

Devido a esses fatores, aumenta a dificuldade no acompanhamento das atividades a serem desenvolvidas, dificulta a forma de medida, compromete o planeamento elaborado, perdendo assim todas as referências de feedback entre o gestor do projeto e os clientes.

Outro problema da gestão de projetos de software está em estimar os prazos e os custos para o projeto. Algumas técnicas auxiliam o gestor do projeto a fornecer valores aproximados de custo e prazo da execução de um projeto de software ao longo do tempo, mas contabilizá-lo no meio de sua execução é uma atividade que requer bom domínio das práticas de gestão do projeto e principalmente que as atividades especificadas para a execução do projeto estejam sendo atualizadas pelos colaboradores. No entanto isso fica mais complicado quando ocorrem mudanças dos requisitos no meio do projeto em execução, impactando diretamente nos prazos e custos planeados inicialmente.

Ao longo deste estudo inúmeras mudanças foram solicitadas para o desenvolvimento deste projeto. Tais mudanças tinham uma enorme influência no planeamento desenvolvido para as atividades definidas anteriormente, comprometendo diretamente os prazos e custos estimados até então.

Diante da necessidade de se avaliar os impactos de mudanças solicitadas em meio ao desenvolvimento do projeto e o acompanhamento de atividades realizadas, a construção de uma ferramenta de software que mostre o andamento das atividades realizadas e aponte os impactos de mudanças no âmbito do projeto atual auxiliando na tomada de decisão para as novas ações de gestão de projetos tornam-se necessárias. Os próximos itens desta seção apresentam a criação deste sistema.

4.2 Implementação

Diante da necessidade de se obter informações rápidas sobre custos, prazos e o controle das atividades, tanto para mudanças de requisitos quanto para novos projetos, o desenvolvimento da solução de software necessitou de uma rápida construção e implantação para que pudesse sanar as imediatas necessidades de controlo dos projetos a serem gerenciados.

Inicialmente foi adotada a metodologia interativa e incremental para a engenharia de software do sistema, pois à medida que os requisitos principais iam sendo finalizados, os mesmos iam sendo disponibilizados aos utilizadores para a gestão de projetos. Com o auxílio da linguagem BPML, pode-se traçar o perfil do processo de negócio quanto ao desenvolvimento de sistema quanto às fases subsequentes do processo. E por fim, para cada requisito solicitado, o mesmo era desenvolvido obedecendo às fases de desenvolvimento de software como apresentado na figura.

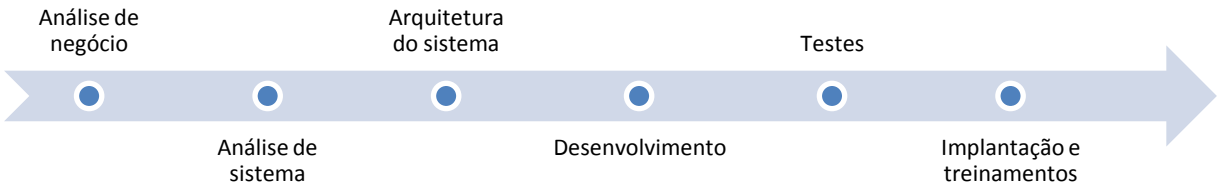


Figura 17 – Fases do processo de desenvolvimento do sistema

Cada uma das fases apresentaram características importantes para o desenvolvimento do modelo de software proposto. Abaixo segue cada uma das características importantes para as fases de desenvolvimento apresentadas.

4.2.1 Análise de negócio

A primeira fase do processo de desenvolvimento de sistemas estabelece os requisitos de negócio para que seja alinhado com o sistema a ser desenvolvido. Foi nesta fase que todas as necessidades foram

tratadas junto ao cliente e analisadas as que de fato iriam agregar valor para a empresa. A Figura 18 representa o processo de Análise de negócio em BPML, sendo possível verificar que deste processo resulta o documento de visão apresentados no anexo I.

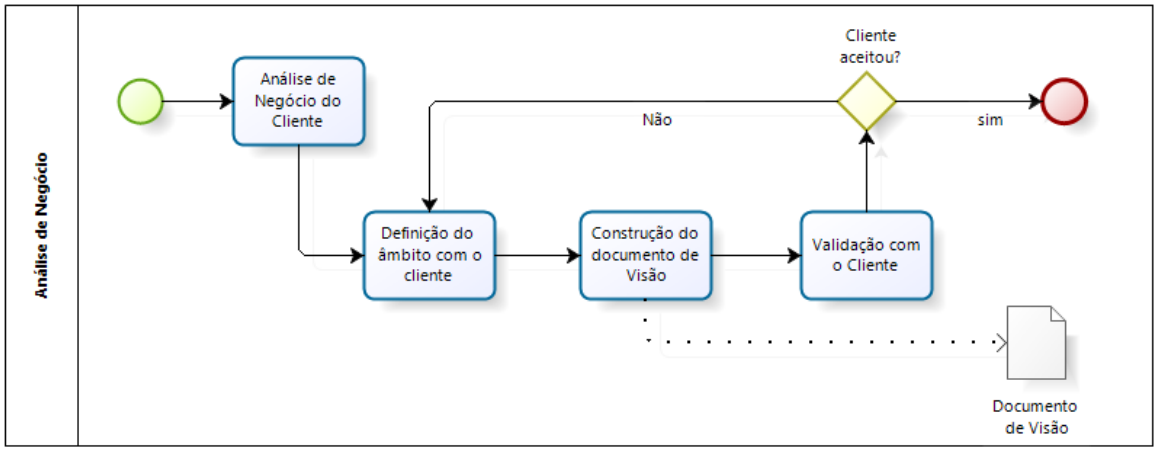


Figura 18 – Diagrama BPML do Processo "Análise de Negócio"

Com base nessas informações pôde-se estabelecer um direcionamento para o projeto solicitado pelo cliente, incluindo a princípio os custos e prazos estimados para se ter a visibilidade da viabilidade do projeto. Essas informações estabelecidas podem sofrer ajustes ao longo do projeto, no entanto não aceitam mudanças que impactem no escopo da aplicação, ou seja, a listagem do requisitos definidos nessa fase são levados até fim do projeto.

4.2.2 Análise de sistema

Após a definição do documento de visão na análise de negócio, a análise de sistemas define os modelos de desenvolvimento do sistema quanto às especificações dos requisitos funcionais, como nos anexos III, IV V e VI, diagramas de classe, sequência, atividades, caso de uso e estados. A Figura 19 apresenta do diagrama de caso para adição de usuário na aplicação no decorrer do funcionamento da aplicação.

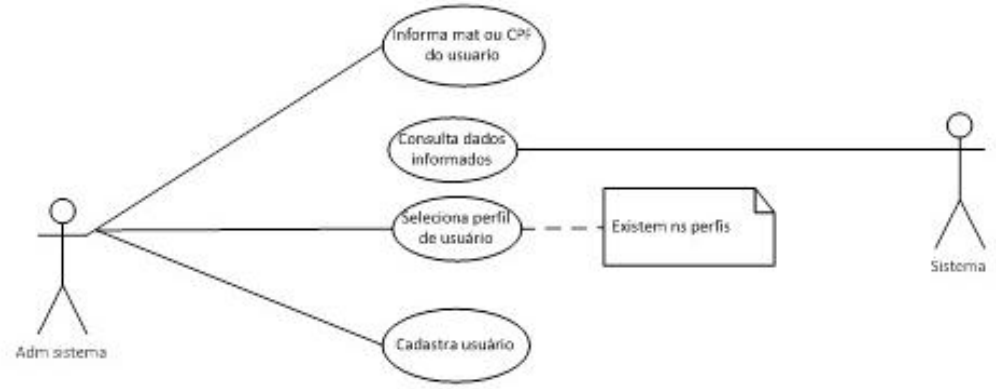


Figura 19 – Diagrama de caso de uso de adição de usuário

A Figura 20 exibe as iterações do caso de uso entre o utilizador e o sistema quanto à adição de um novo utilizador cadastrado a uma nova funcionalidade de sistema. Este diagrama representa de forma simbólica o fluxo principal descrito de forma textual no documento de especificação de requisitos (Ver anexo III). O diagrama também informa as tomadas de decisões e alguns tratamentos de exceções.

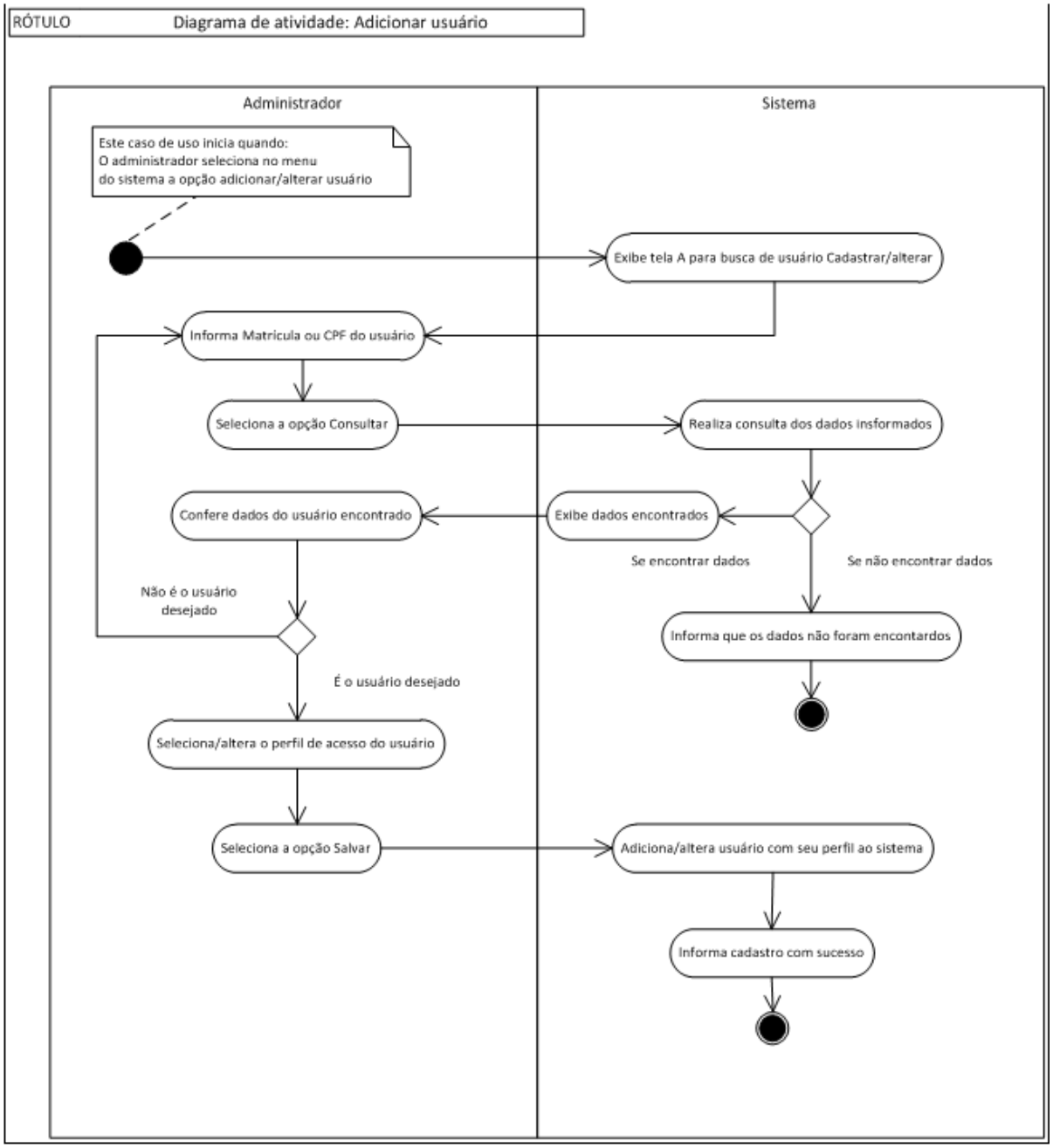


Figura 20 – Diagrama de atividade de adição de usuário

A Figura 21 apresenta as iterações dos objetos instanciados na aplicação desenvolvida e as informações das entidades que persistem na base de dados. Esse diagrama reflete as entidades a

serem desenvolvidas no banco de dados, no entanto os métodos que a mesma apresenta ficam disponíveis apenas na camada de aplicação e executadas à medida que o utilizador manipula informações no sistema.

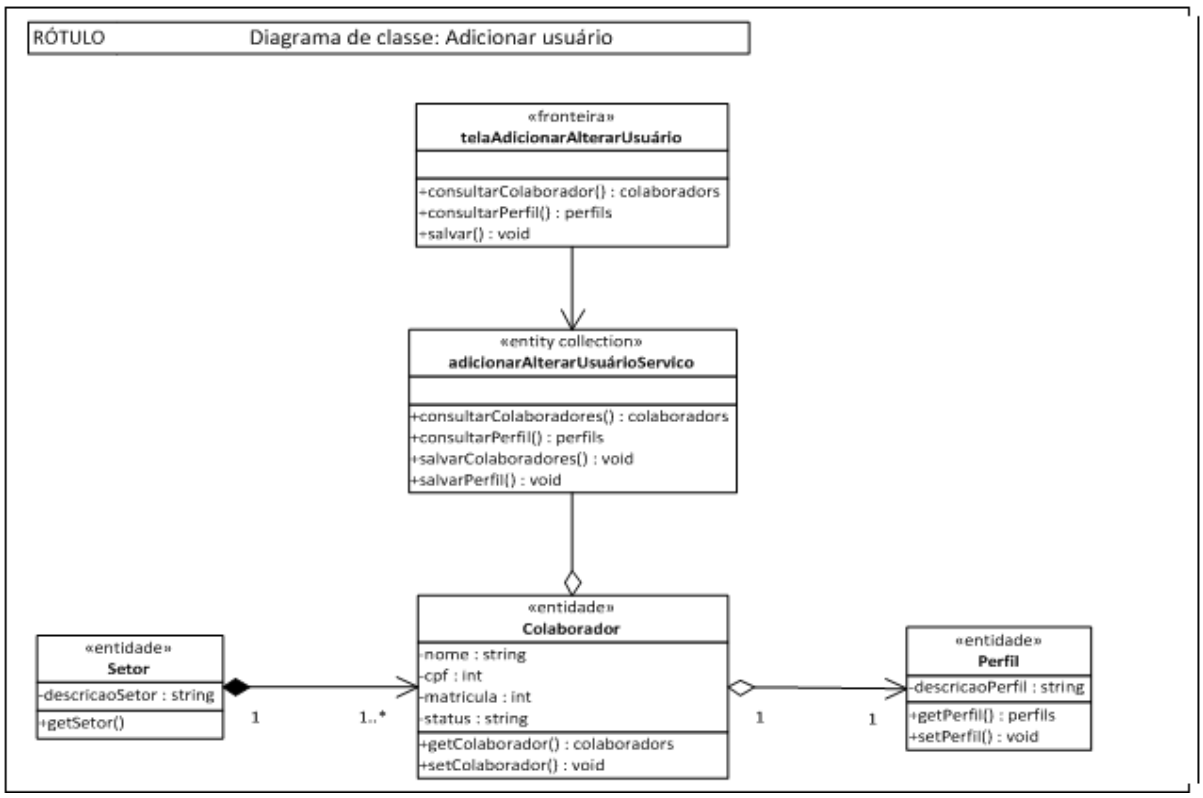


Figura 21 – Diagrama de classe da função de adição de usuário

A Figura 22 exibe a iteração dos métodos inseridos nas camadas de aplicação do sistema à medida que os mesmos são executados (manipulada informações pelo utilizador) no sistema. O diagrama exibe as camadas de procedimentos à medida que as iterações entre as classes são chamadas na aplicação.

Nesta fase também foram definidas as regras de negócio quando ao comportamento do sistema, utilizadores, pré e pós-condições quanto à utilização do sistema homologado em um documento elaborado junto ao cliente. Os diagramas de caso de uso e os documentos de especificação de caso de uso encontram-se nos anexos III e IV deste documento respetivamente.

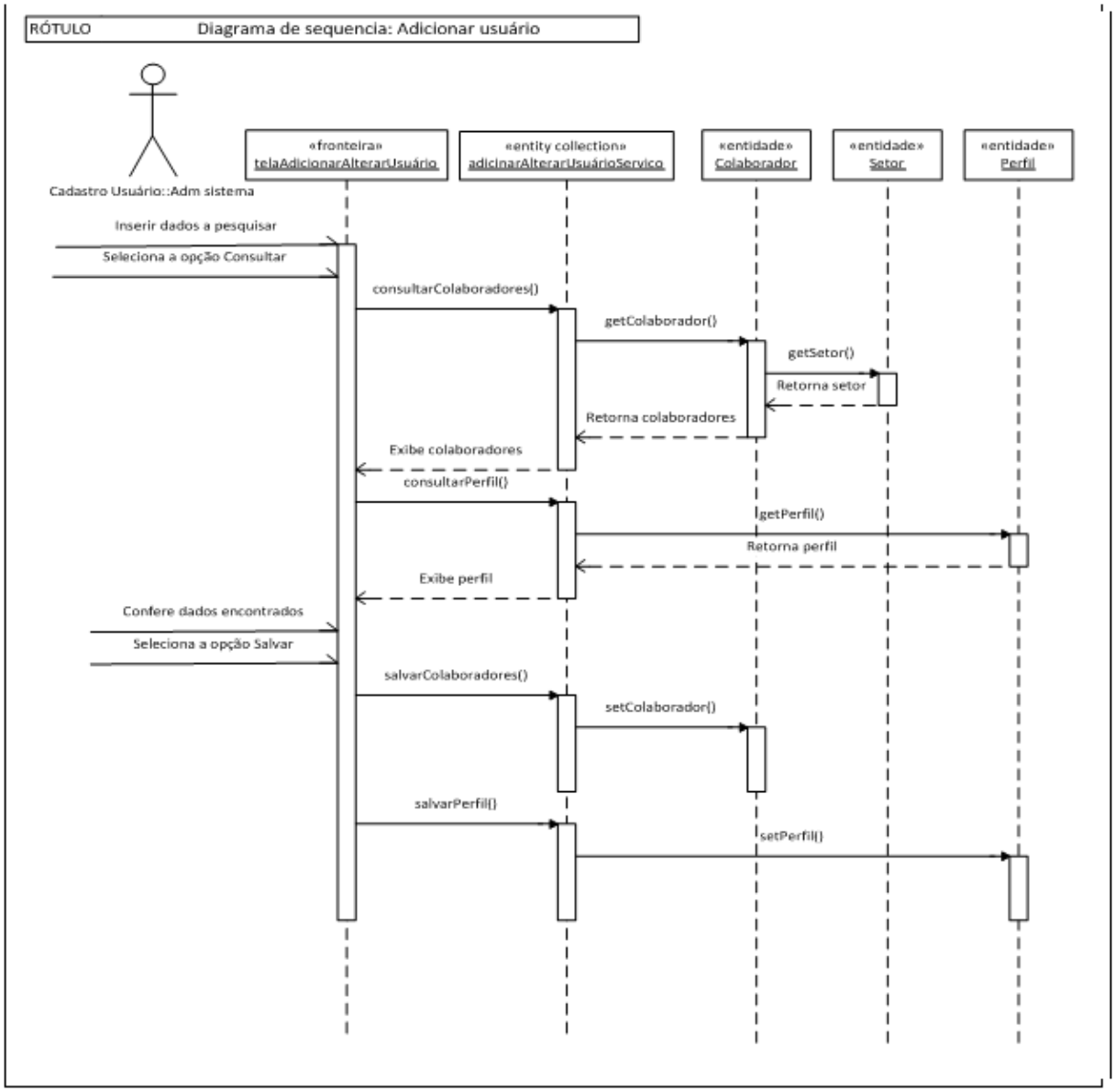


Figura 22 – Diagrama de sequencia de adição de usuário

Por fim a Figura 23 representa o processo de “Análise de sistemas” em BPML, sendo possível verificar que deste processo resulta o diagrama de caso de uso, modelo entidade relacional e documento de especificação de requisitos.

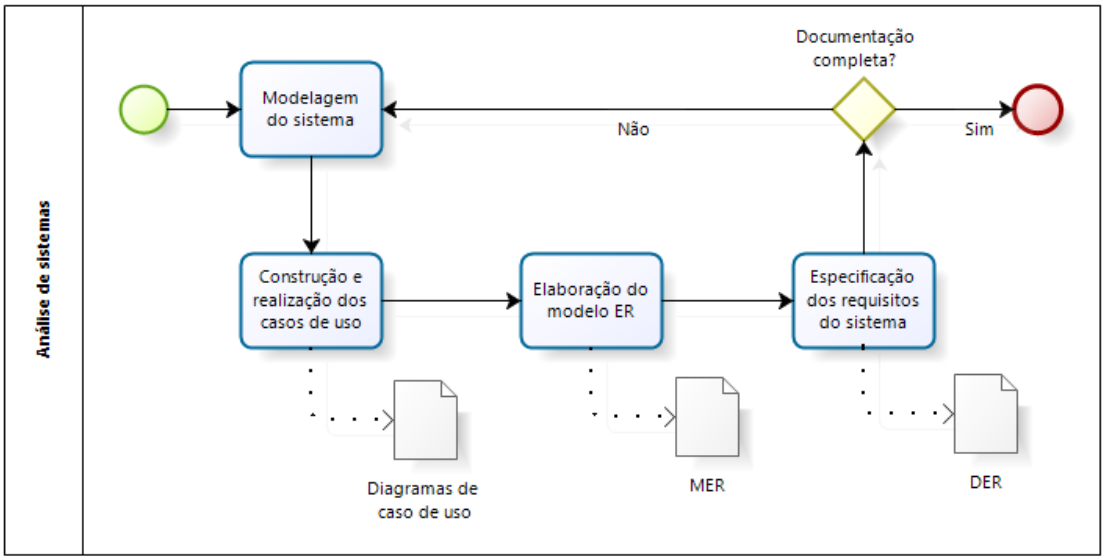


Figura 23 – Processo de Análise de sistemas em BPML

4.2.3 Arquitetura do sistema

Na fase de arquitetura de sistemas contempla o desenvolvimento da estrutura do sistema quanto a tecnologia a ser utilizada, a definição das bibliotecas de desenvolvimento e a construção do modelo de dados da aplicação. A Figura 24 representa o processo de Arquitetura de sistemas em BPML, sendo possível verificar no final deste processo a estrutura do código fonte da aplicação e o modelo de dados implementado.

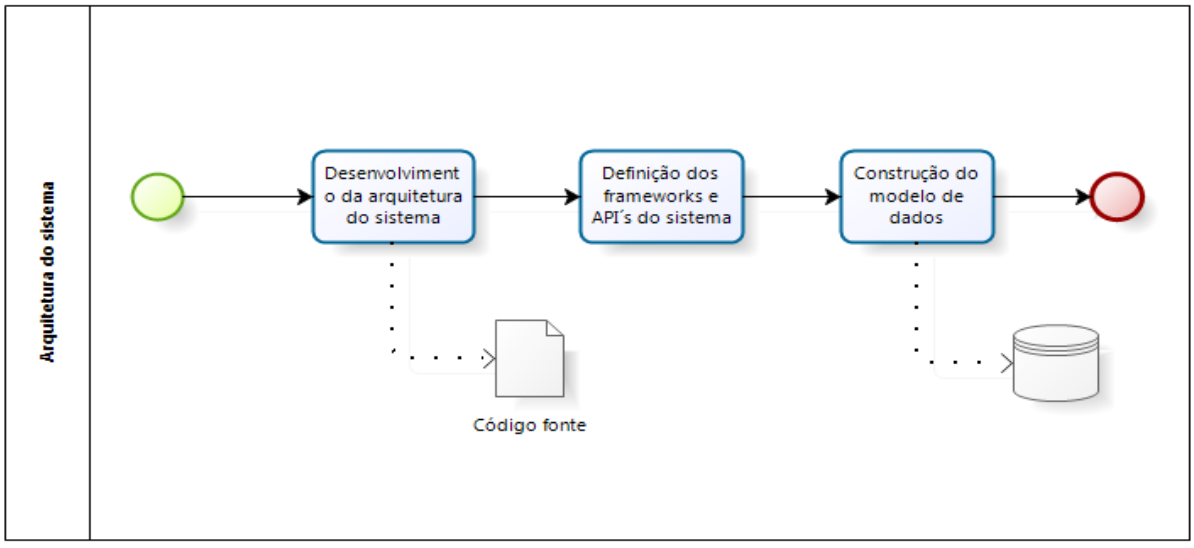


Figura 24 – Processo de Arquitetura de sistemas em BPML

O sistema de gestão projetos de software teve sua estrutura de camadas de acesso a dados, criação do bando de dados e definição das linguagens de programação nesta fase. A Figura 25 apresenta a divisão do sistema desenvolvido em suas respetivas camadas para cada uma das funcionalidades

solicitadas e descritas nos requisitos e com a camada DAO (Data Access Object) sendo a referência de acesso e controle a dados. As mesmas utilizam as tecnologias disponíveis no framework Net 4.0.

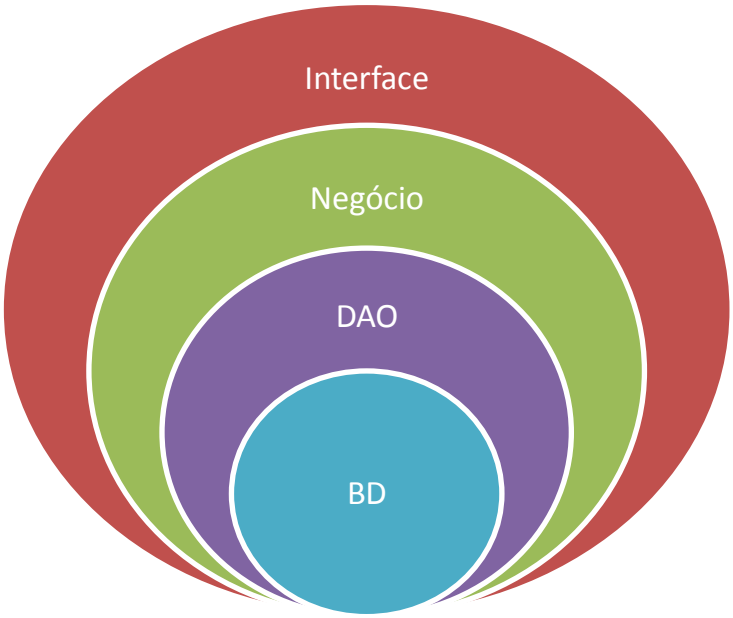


Figura 25 – Arquitetura de camadas do sistema

Da parte externa para a interna, o sistema se inicia pela camada de interface que é a iteração do utilizador com o sistema. É através dessa camada que as requisições de utilizador, ações e eventos externos do sistema são solicitadas. Em seguida, a camada de negócio é responsável pelas validações necessárias do sistema junto aos eventos e solicitações requeridos através da camada de interface. É esta camada especificamente que muda de uma aplicação de rotinas financeiras para uma aplicação de rotinas contábeis por exemplo. É diretamente na camada de negócio que os requisitos definidos junto ao cliente devem ser implementados juntamente com as regras de negócio. A camada DAO é responsável pelos métodos de acesso, consultas e demais manipulações de dados. Todas as rotinas requeridas pela camada de interface que foram devidamente validas na camada de negócio ao chegarem na camada DAO identificam os respetivos métodos requeridos para retornar uma resposta ao utilizador solicitante. Por fim na camada de BD (*Data Base*) é responsável pelo armazenamento das informações inseridas e/ou manipuladas pelos processos de negócio do sistema.

A Figura 26 apresenta a camada DAO, em modo de projeto, com seus respetivos métodos de acesso a dados retratados na classe pública BLL (*Business Logic Layer*) que é responsável pelas operações básicas em uma base de dados, consulta, insere, altera e exclui.

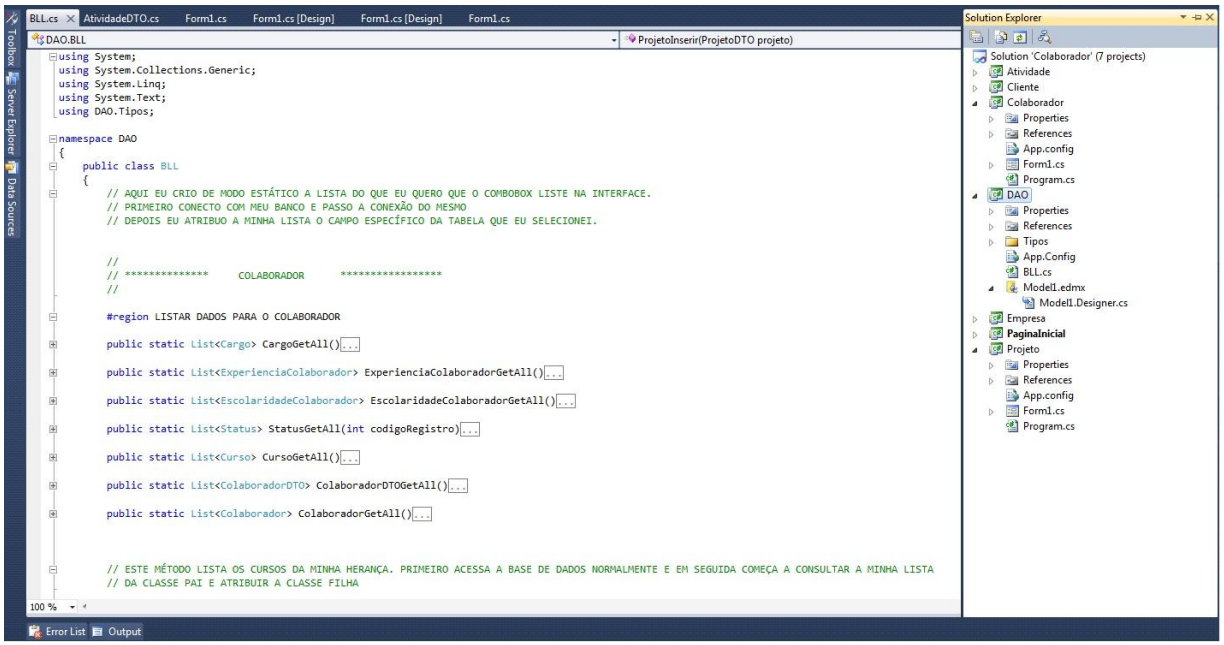


Figura 26 – Métodos da camada DAO do sistema desenvolvido

Cada um dos métodos apresentados na camada DAO é solicitado nas respetivas camadas de negócio do sistema, à medida que o mesmo é solicitado pela aplicação.

A seguir verifica-se o modelo de banco de dados desenvolvido para esta aplicação, bem como seus relacionamentos com as entidades construídas a partir das necessidades do sistema definidos no documento de especificação de requisitos utilizando o sistema de gerenciamento de banco de dado SQL Server 2008 R2. A Figura 27 apresenta uma vista parcial do modelo de entidade-relacionamento implementado neste sistema.

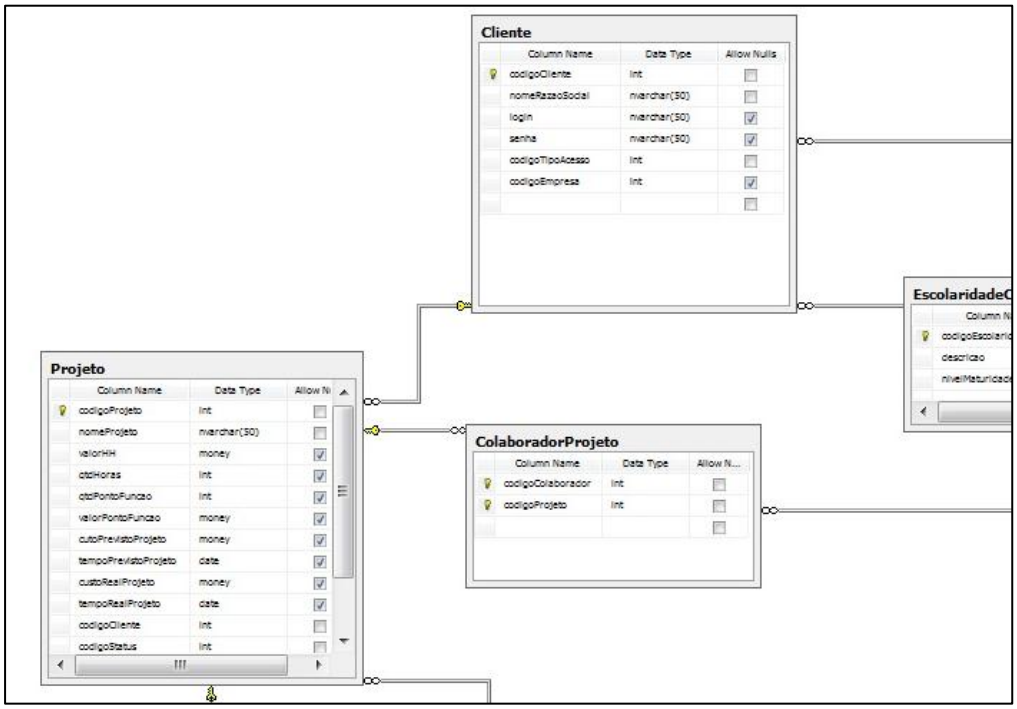


Figura 27 – Modelo de dados em diagrama entidade relacional do sistema

4.2.4 Desenvolvimento

Após definida e construída a estrutura do sistema, o processo parte para a fase de desenvolvimento de acordo com os itens especificados nas documentações resultantes da fase de análise de sistemas. A Figura 28 representa o processo de Desenvolvimento em BPML.

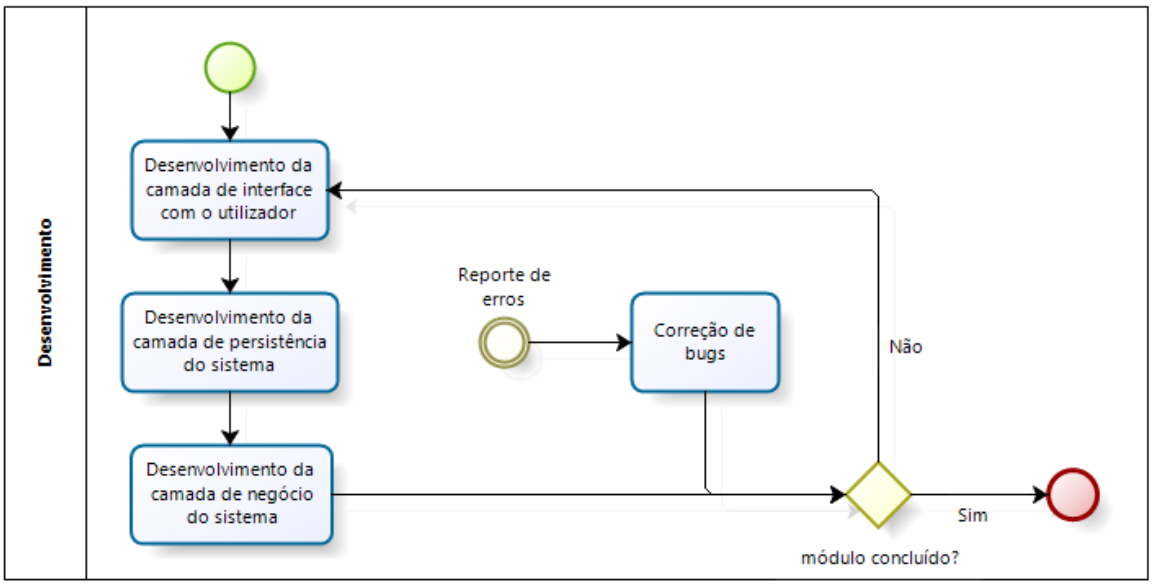


Figura 28 – Processo de Desenvolvimento em BPML

Após toda a arquitetura do sistema definida e os requisitos do sistema totalmente documentados e homologados junto ao cliente, o sistema teve a sua conceção dividida em duas partes. Na primeira o desenvolvimento do designe da interface do utilizador com o sistema, tanto nas consultas de dados quanto na manutenção dos mesmos como mostra a Figura 29.

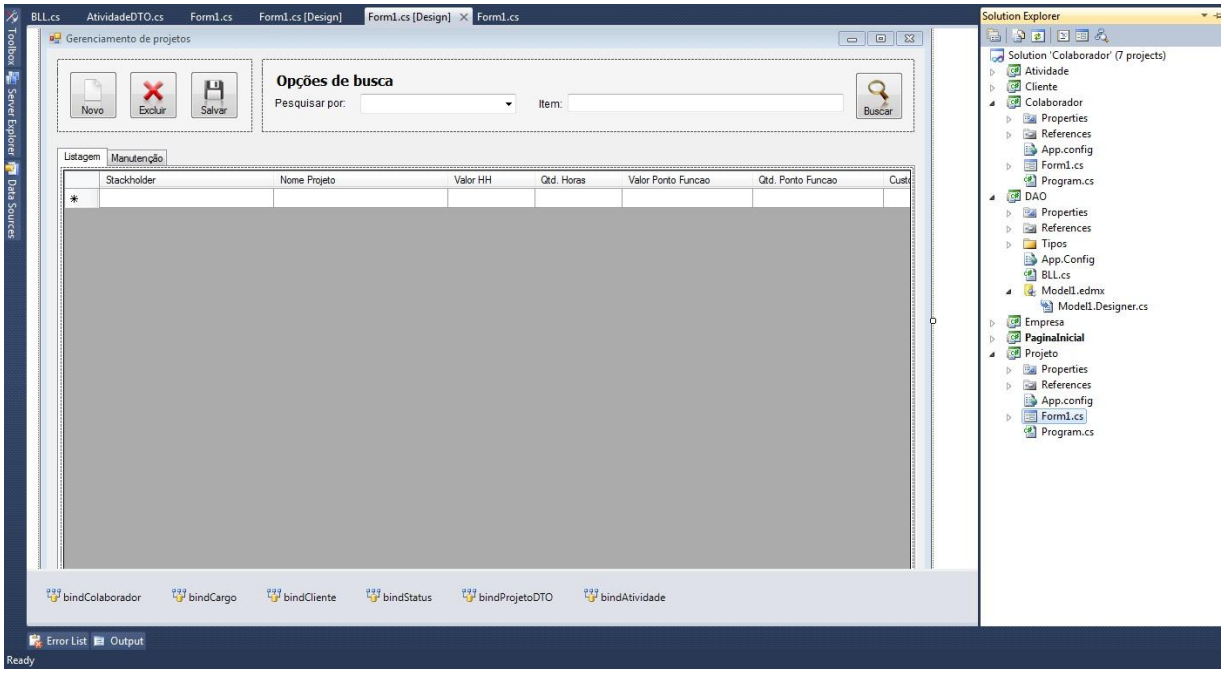


Figura 29 – Camada interface (Apresentação) do sistema – Lista de projetos

Na Figura 30 é possível realizar o desenvolvimento de interfaces que viabilizam a manutenção dos dados listados na figura anterior. Com o vasto poder de desenvolvimento promovido no ambiente de interface, a navegabilidade do sistema é bem definida.

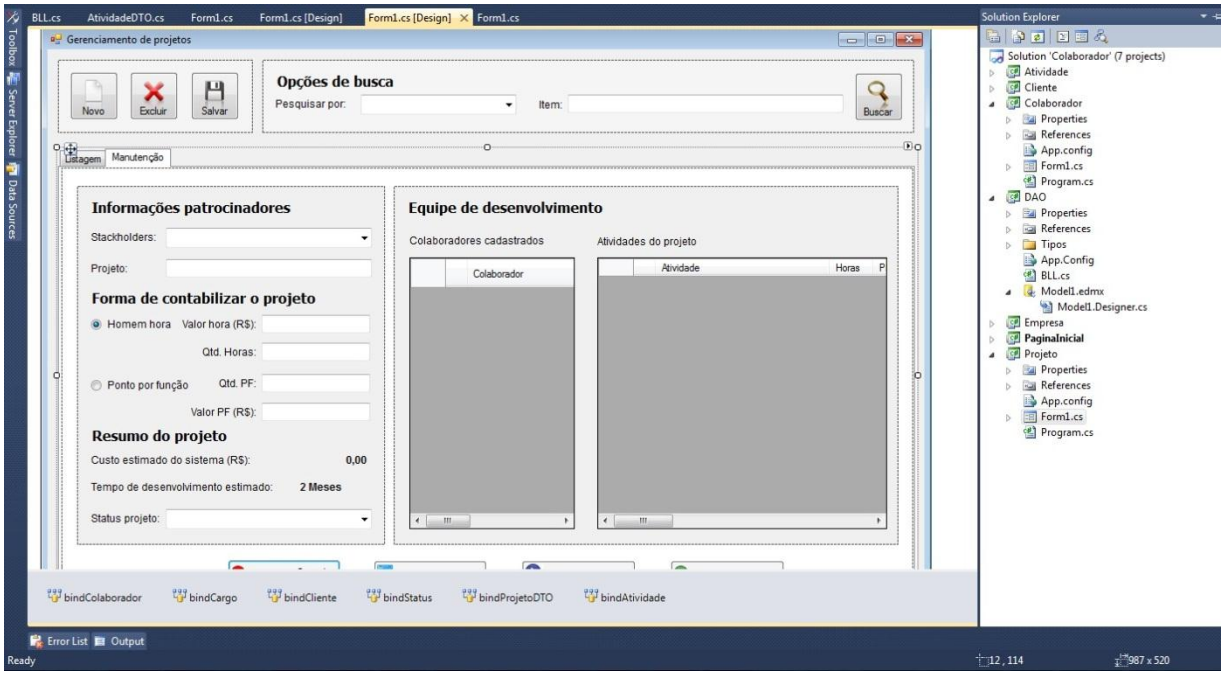


Figura 30 – Camada interface (Apresentação) do sistema – Manutenção de projetos

Por fim, a camada de negócio do sistema que efetua as validações necessárias e comunica a interface ao acesso a dados como mostrado na Figura 31.

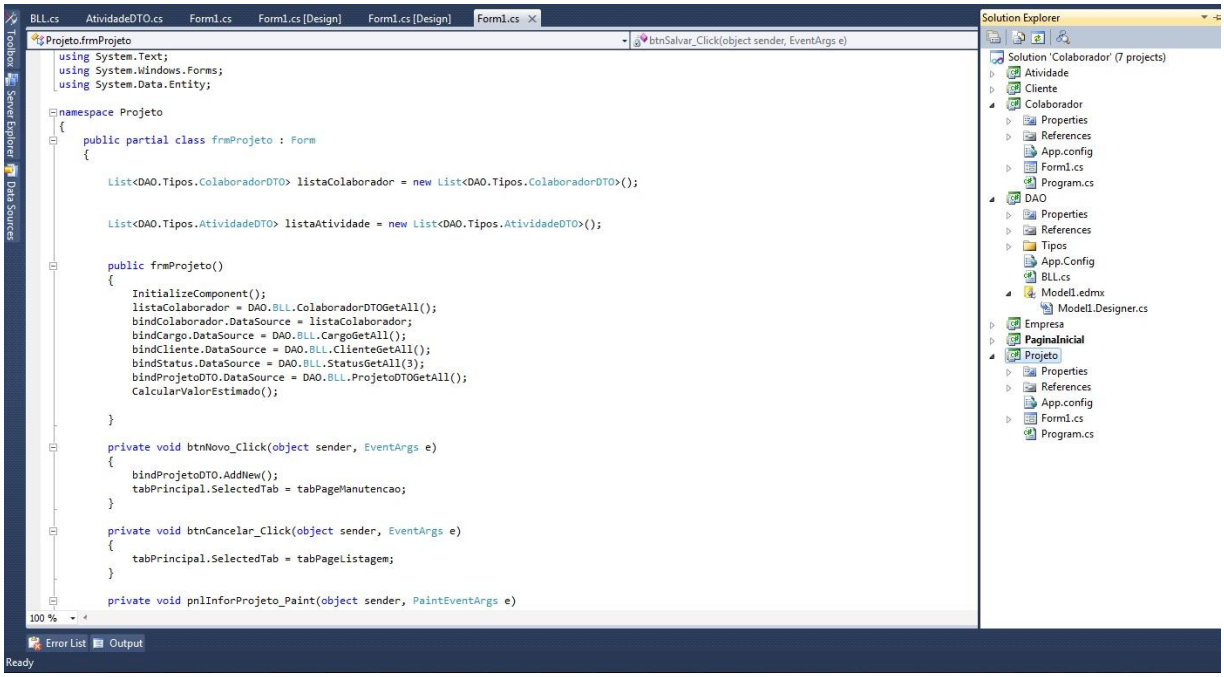


Figura 31 – Camada de negócio da aplicação

Toda a conceção da aplicação desenvolvida foi realizada utilizando a linguagem C Sharp, que juntamente com o framework Dot Net 4.0 facilitou as atividades de desenvolvimento e construção das funcionalidades solicitadas.

Após o desenvolvimento de cada um dos módulos e suas respetivas funcionalidades (Interfaces e Camadas de Negócio), o desenvolvedor realiza a integração das funcionalidades desenvolvidas com a arquitetura proposta, realizando assim, por parte da aplicação, os respetivos acessos a dados de acordo com a funcionalidade desenvolvida.

À medida que novos módulos forem sendo desenvolvidos, novas integrações serão necessárias para que as funcionalidades executem todas as rotinas requeridas.

4.2.5 Testes

Após o desenvolvimento da aplicação, a mesma é enviada a fase de teste para validação dos requisitos estabeleciodos pelo cliente, podendo esta fase reportar erros a fase de desenvolvimento ou encaminhar a aplicação desenvolvida para a fase de implantação. A Figura 32 apresenta o processo de Teste em BPML.

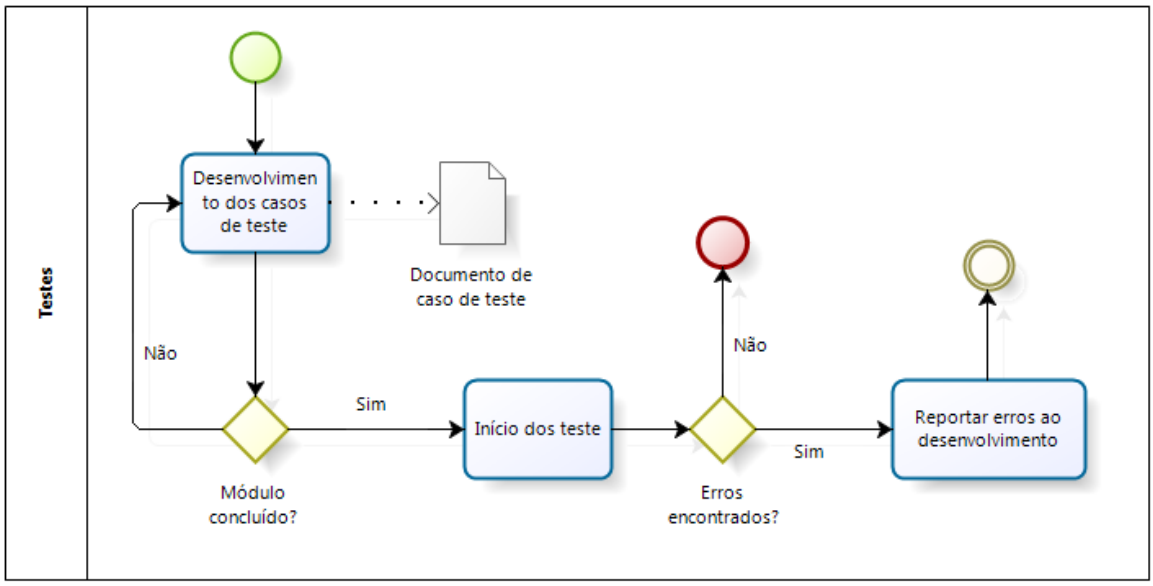


Figura 32 – Processo de Teste em BPML

A cada funcionalidade desenvolvida, a mesma era disponibilizada para a fase de teste, que tinha o principal objetivo de garantir que as necessidades exigidas para o sistema fossem de fato desenvolvidas de acordo com as especificações dos requisitos do documento. Os testes simulavam o

ambiente de gerenciamento de projetos de software, bem como a gestão das atividades solicitadas, a consulta de valores previstos para os custos e prazos tanto para novos projetos quanto para mudanças solicitadas para os projetos em andamento. Alguns bugs (não conformidades) foram detetados e resolvidos pelos programadores logo que identificados.

4.2.6 Implantação e treinamentos

Por fim a fase de implantação homologa junto ao cliente as entregas realizadas de acordo com o seus relatórios de implantação e realização de treinamentos. A Figura 33 apresenta o processo de implantação e treinamento em BPML.

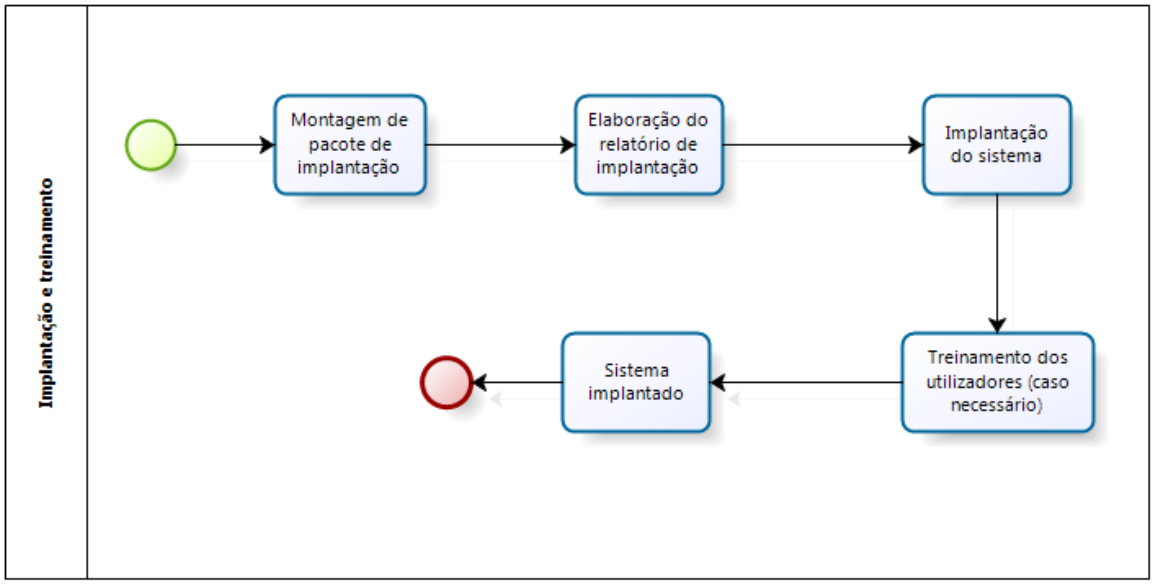


Figura 33 – Processo de Implantação e Treinamento em BPML

No diagrama de implantação da Figura 34 exibe as iterações entre as camadas de aplicações desenvolvidas, as aplicações que executam no servidor e por fim as entidades dispostas no banco de dados do sistema.

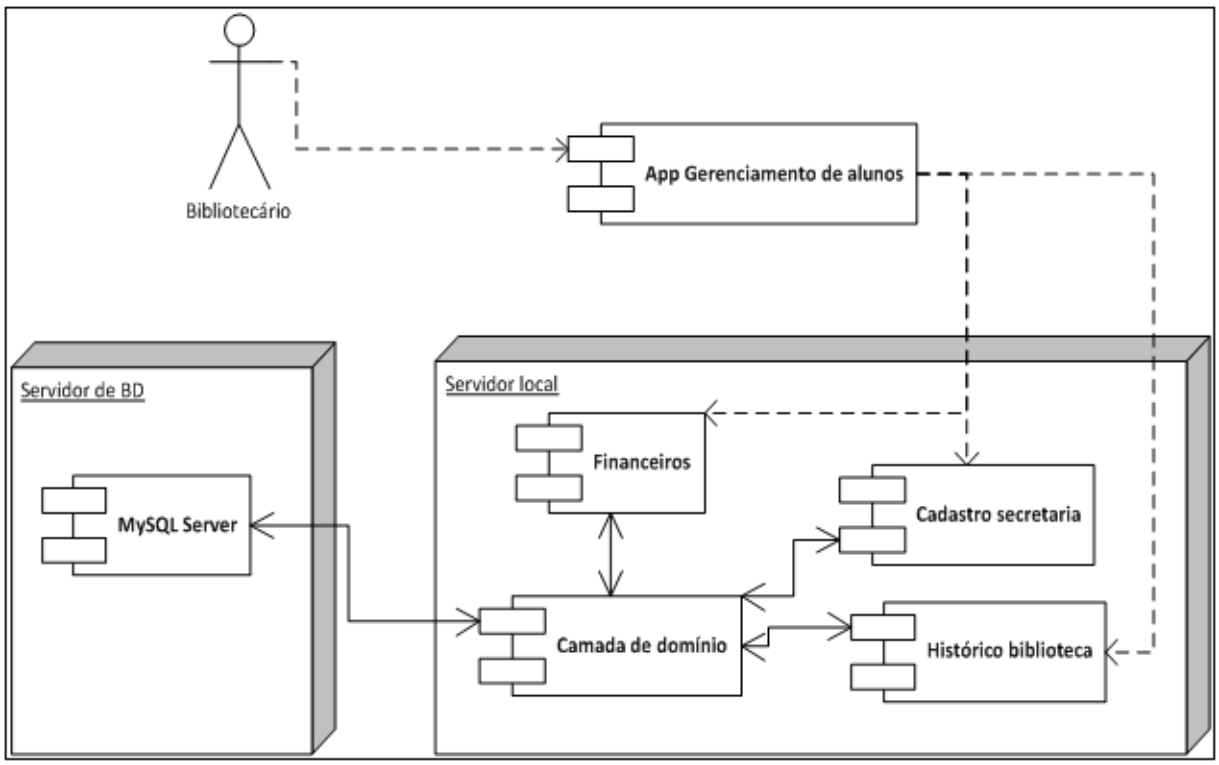


Figura 34 – Diagrama de implantação do sistema

Após a finalização do sistema, o mesmo requereu apenas instruções básicas aos colaboradores do projeto para terem acesso às suas atividades, visualizá-las, iniciá-las e finalizá-las. Finalizada essas atividades, a parte de acompanhamento de custos e prazos com previstos e realizados fica a critério do gestor de projetos da equipe, bem como analisar o andamento das atividades planeadas quanto organizar projetos futuros. O acompanhamento das atividades realizadas é realizado de forma gráfica e por relatórios gerados pelo sistema.

Logos após o desenvolvimento do sistema, os projetos passaram a ter mais visibilidade quanto ao seu acompanhamento de uma maneira muito mais rápida e confiável, permitindo ao gestor de projetos juntamente com o cliente verificarem se poderia ser viável a aplicação de uma mudança nos projetos em andamento ou se compensava elaborar e planejar um novo projeto a se realizar sem comprometer os recursos empregados nos projetos atuais.

O sistema também proporcionou mais dinamismo ao acompanhamento das atividades possibilitando, através de uma plataforma web, a realização das atividades de forma descentralizada da empresa, onde alguns colaboradores selecionados tinham suas atividades a realizar, porém os mesmo a iniciavam e/ou a finalizavam de qualquer ponto de acesso a internet, fazendo com que o acompanhamento projeto tenha mais fiabilidade nas informações.

4.3 Resultados

Contudo, após o período de pesquisa, análise do problema, implementação e implantação do sistema de gestão de projeto de software, resultados relevantes foram identificados no acompanhamento de atividades referente ao projeto e apoio a tomada de decisão por parte do gestor, como listados a seguir:

- A organização em listagem dos projetos em desenvolvimento paralelo pela equipe a medida que as atividades são necessárias para o desenvolvimento do mesmo. A Figura 35 exibe a interface de lista de projetos cadastrados para serem executados ou que estão em execução.

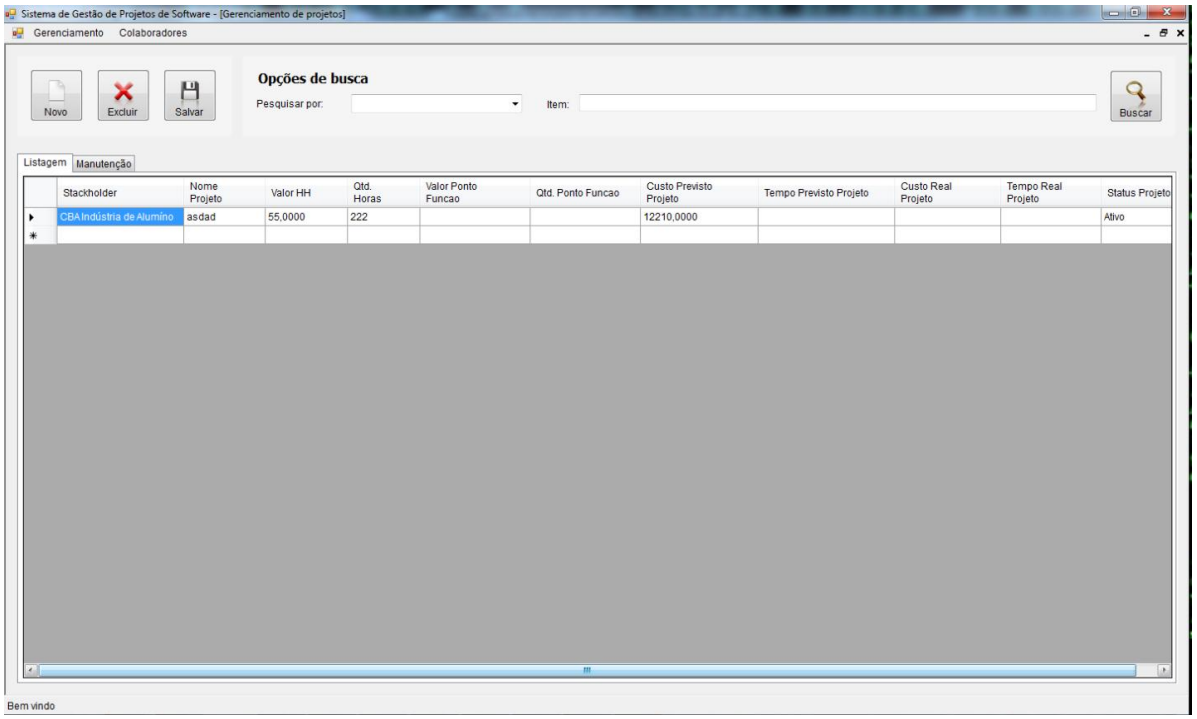


Figura 35 – Lista principal de acompanhamento de projetos

- O cadastro de projetos mediante as necessidades dos clientes, a alocação dos recursos pessoais disponíveis e a visualização dos prazos dos custos necessários para a realização do projeto antes de iniciá-lo. A seguir, a Figura 36 exibe os detalhes do projeto cadastrado no sistema que permite o cadastro de um novo projeto, a descrição dos clientes do projeto, os colaboradores disponíveis juntamente com suas respetivas atividades e principalmente o cálculo de custos previsto para o desenvolvimento do projeto.

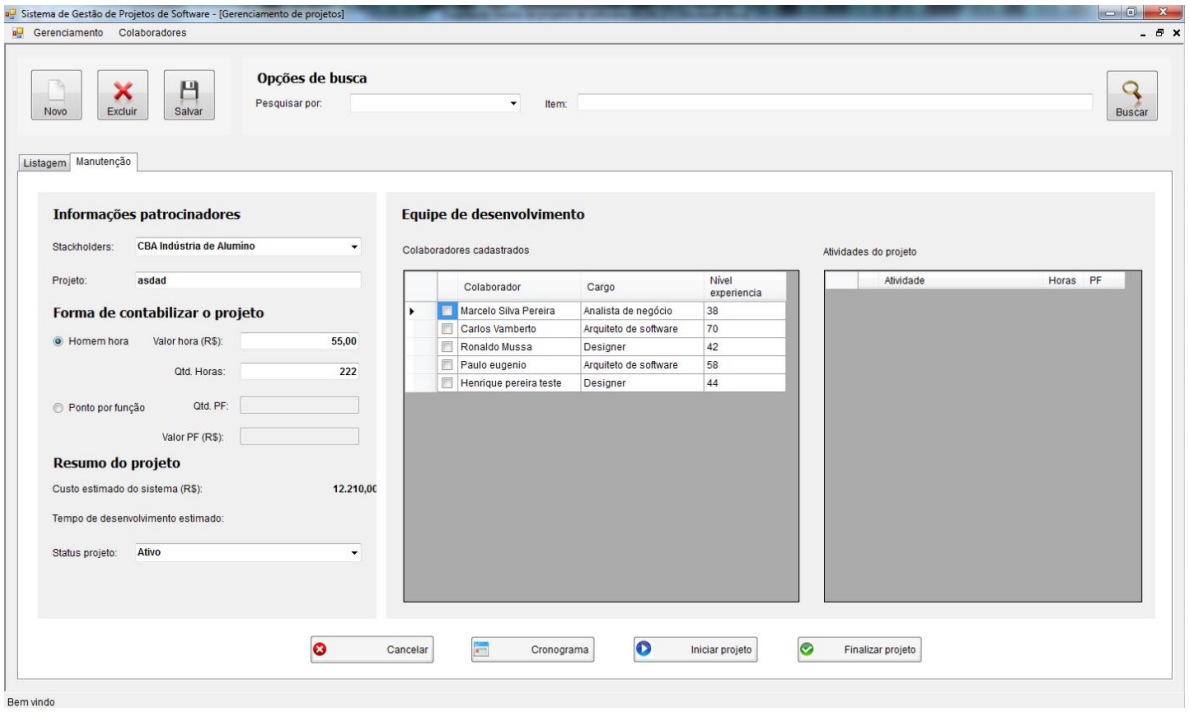


Figura 36 – Interface de cadastro de projetos

- O cadastro dos recursos pessoais de acordo com a formação, tempo de experiência e o aperfeiçoamento contínuo do colaborador, definindo assim um nível de experiência que impacta no tempo de desenvolvimento das atividades atribuídas para um determinado projeto posteriormente à alocação do mesmo no projeto. A Figura 37 demonstra a interface do cadastro de colaborador bem como sua formação, experiência e certificações.

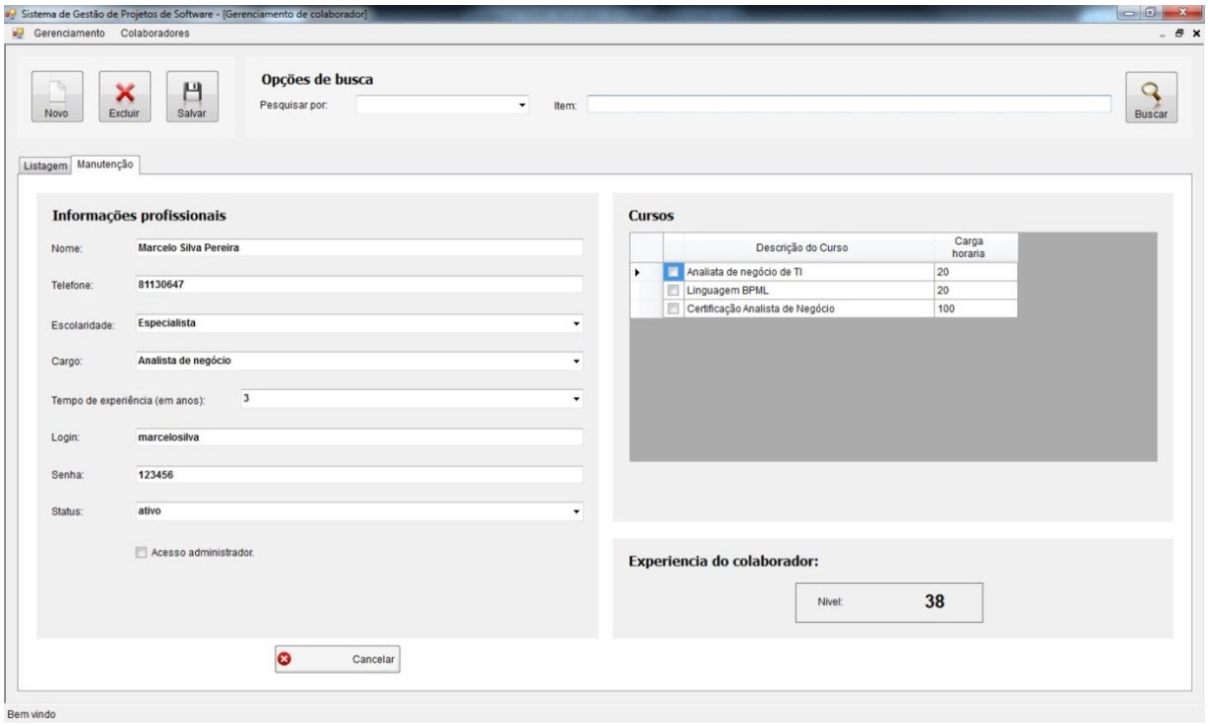


Figura 37 – Interface de cadastro de colaboradores

- O acompanhamento das atividades em desenvolvimento referentes ao projeto pelo período estabelecido inicialmente no seu cadastro e suas respetivas pendencias em tempo real. A Figura 38 demonstra o acompanhamento das atividades do projeto sol diamante (utilizado como piloto), o status das atividades em andamento destacadas em cores e os respectivos nomes dos recursos ao lado da mesma.

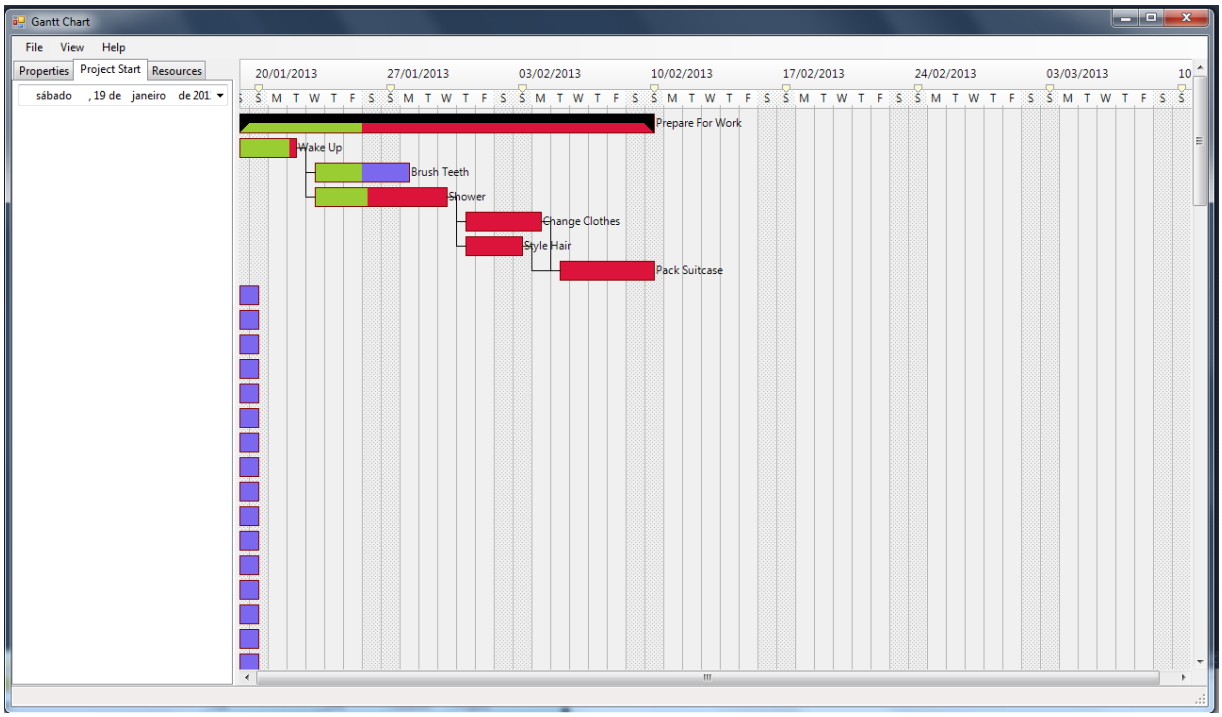


Figura 38 – Acompanhamento de projetos em desenvolvimento

- E por fim a análise do impacto dos prazos e custos reais e estimados relacionados à mudança de requisitos perca ou adição de recursos e alteração dos prazos estabelecidos com relação aos mesmos parâmetros executados realmente. A mesma análise também apoia na visualização de fases que necessitam de melhorias em seu processo de desenvolvimento.

Com essas informações a tomada de decisão se torna mais rápida para definir se há viabilidade de desenvolvimento de um novo projeto de software de acordo com os recursos de colaboradores disponíveis, no tempo especificado e principalmente com os custos apresentados com o objetivo de melhorar sempre as fases do processo de desenvolvimento dos sistemas, identificando gargalos e assim contribuir para melhorar o negócio de produção de software da empresa.

Após as respetivas análises (documentações e modelagens), implementações, testes e treinamentos, o sistema começou a ser utilizado na empresa em modo experimental pelos utilizadores colaboradores (membros da equipe do projeto em desenvolvimento), para fixar a rotina de iniciar e finalizar tarefas

realizadas referentes ao projeto de software que estava ativo. A princípio, atividades comuns aos projetos foram cadastradas pelo gestor do projeto bem como os tempos estimados para execução da atividade baseado em um nível de serviço de mercado de desenvolvimento de sistemas. As tarefas iniciadas e finalizadas de modo experimental colaboravam para o acompanhamento do desenvolvimento do projeto, o que depois veio proporcionar o acompanhamento de prazos e custos previstos e reais dos sistemas em questão.

Finalizada a fase experimental, o primeiro projeto a ser gerenciado pelo sistema foi o módulo contas a pagar e receber do sistema Sol Diamante, que por sua vez gerou várias tarefas a serem realizadas, a alocação de colaboradores com suas respetivas atividades, prazos e custos estimados. Ao iniciar o projeto, o mesmo apresentava um cronograma previsto para o prazo a ser executado e um cronograma secundário para as atividades realmente realizadas, este acompanhamento real era atualizado a medida que os colaboradores do projeto iam finalizando as tarefas iniciadas.

À medida que as atividades eram iniciadas e finalizadas o sistema atualizava o cronograma juntamente com o prazo que ainda cabia ao projeto e seu respetivo custo real até o momento em que foi desenvolvido. Por sua vez, uma mudança foi solicitada em pleno desenvolvimento do projeto, o que possibilitou o sistema a auxiliar em um ponto fundamental, na tomada de decisão do gestor do projeto. Verificou-se que a mudança traria impactos relativamente grandes no custo e no prazo do projeto que já estava em desenvolvimento, o que possibilitou a conclusão de que a mudança solicitada seria mais conveniente ser desenvolvido em um segundo momento. Esse auxílio para a tomada de decisão por parte da equipe gestora do projeto, contribui grandiosamente para a otimização dos prazos, custos e recursos de mão-de-obra ao longo do projeto.

5 CONCLUSÃO

Este estudo teve como objetivo desenvolver o protótipo de um sistema que apoie o acompanhamento do processo de produção de um sistema informático mostrando o estado atual do desenvolvimento do sistema, relação entre custos e tempo de construção e comparação entre tempo proposto para produção e tempo realmente utilizado que sirvam de apoio à tomada de decisão para agilizar o desenvolvimento do projeto de software.

Na execução deste trabalho foi-se realizando o acompanhamento dos processos de negócio de uma empresa de desenvolvimento de sistemas que era o objeto de nosso estudo e de acordo com seus processos foi-se mapeando o processo que a mesma executava para a realização de suas atividades. Após essa etapa, foi-se abordando então junto à equipa questionamentos sobre o que se poderia melhorar no processo de desenvolvimento de sistemas e principalmente como geri-lo de maneira mais efetiva, acompanhando e analisando seu desenvolvimento ao longo do tempo alinhado as técnicas de gestão de projetos.

Durante o processo de desenvolvimento dessas ações, foi possível acompanhar a execução de atividades referentes ao planeamento de novos projetos de software, a execução das atividades referentes à construção dos sistemas (camadas de negócio, camadas de acesso a dados, modelagem de banco de dados entre outros elementos da aplicação em desenvolvimento), a realização de testes funcionais, a integração/implantação dos sistemas para o ambiente de produção, o treinamento junto aos utilizadores e por fim o acompanhamento da gestão desses projetos.

Diante das informações observadas e analisadas foi possível modelar, em uma linguagem de modelagem de processo de negócio (no caso BPML), as fases do processo de desenvolvimento de sistemas que deveriam ser adotados e com isso informações necessárias para implementar um sistema para apoio à gestão de projetos de software.

Deve-se destacar que o resultado deste estudo não proporcionou apenas mais uma ferramenta de gestão de projetos específica para sistemas de software, mas sim uma ferramenta que auxilia as tomadas de decisão de um ou mais gestores diante de necessidades de mudanças ou novos projetos a serem desenvolvidos de acordo com os requisitos do cliente, como também identificar pontos de melhoria nos processos de construção de sistemas ou até mesmo gargalos de colaboradores responsáveis por desenvolver partes do processo.

Com o passar do tempo e maior inserção de dados no sistema, o mesmo poderá apresentar informações mais fidedignas a realidade da equipe de desenvolvimento de sistemas, tanto quanto as suas limitações quanto ao seu tempo de resposta a execução de uma atividade ou até mesmo uma fase inteira do processo de desenvolvimento de sistemas. Para isso novas funcionalidades para a

análise de novas informações para a gestão de projetos de software sendo construído em um segundo momento deste estudo pode ampliar a pesquisa em relação à informações a se analisar, podendo ser esse um ponto de partida para novas pesquisas relacionadas à prática de gestão e controle de atividades referentes a projetos de software.

REFERÊNCIAS BIBLIOGRÁFICAS

Bezerra, Eduardo (2002) Princípios de análise e projeto de sistemas com UML. Rio de Janeiro: Elsevier, 4° Edição.

Blaschek, José Roberto (2009) Gerência de Projetos de Software: Processos, Métodos e Técnicas da Engenharia de Software.

Bortolini, Rafael (2006) Padronizando Processos: BPMN, BPML, XPDL e BPEL. Baguete Tecnologia e informação em um só lugar. Disponível em: www.baguete.com.br/artigosDetalhes.imprime?id. Acessado em 17/10/2012.

Curtis, B. (2001) Hefley, W., Miller, S., People Maturity Compability Model, Version 2.0, CMU/SEI-2001-MM-01, Software Enginnering Institute, Carnegie Mellon University, Pittsburgh, PA.

Dias, R. (2012) "Análise por Pontos de Função: Uma Técnica para Dimensionamento de Sistemas de Informação", on-line. Disponível em: www.presidentekennedy.br/resi/edicao03/artigo02.pdf. Último acesso: 13.03.2012.

Heldman, Kim (2005) Gerência de Projetos: Guia para o exame oficial do PMI, 3ª edição, Editora Campus.

Hoyer, Volker; Bucherer, Eva and Schnabel, Florian (2007) Collaborative e-Business Process Modelling: Transforming Private EPC to Public BPMN Business Process Models; SAP Research CEC St. Gallen, Switzerland Institute for Media and Communication Management, University of St. Gallen, Switzerland.

Engholm, E (2010) Engenharia de Software na Prática: Novatec. São Paulo.

Frame, J. (1994) The New Project Management: Tools for an Age of Rapid Change, Corporate Reengineering, and Other Business Realities, Jossey-Bass Inc., S. Francisco, CA.

IFPUG (2004) Counting Practices Manual. Version 4.2.

Keil, M. (1995) "Pulling the Plug: Software Project Management and the Problem of Project Escalation", MIS Quarterly, Vol. 19, Nr. 4, pp. 421-448.

Kniberg, Henrik (.2007) Scrum e XP Direto das Trincheiras. IBM.

Marques, Paulo; Pedroso, Hernâni, Pedroso; Figueira, Ricardo (2009) C# 3.5, Editora FCA – Editora de Informática, Lda, Lisboa, Portugal.

Mertz, Sharon (2007) Notícias – Negócios: Mercado de software como serviço vai ter forte alta até 2011. Por COMPUTERWORLD 14 de agosto de 2007 - 7h20 http://computerworld.uol.com.br/negocios/2007/08/14/idgnoticia.2007-08-14.0458792221/.

Mertz, Sharon (2012) Gartner Says Worldwide Software-as-a-Service Revenue to Reach $14.5 Billion in 2012, STAMFORD, Conn., March 27, 2012 View All Press Releases http://www.gartner.com/newsroom/id/1963815.

Miguel, Antonio (2010) Gestão de Projectos de Software, 4° Edição, Editora FCA – Editora de Informática, Lda, Lisboa, Portugal.

Minayo, Maria Cecília de Souza (1994) Pesquisa Social: teoria, método e criatividade. Petrópolis: Vozes.

Patterson, David A. & Hennessy, John L. (2000) Organização e projetos de computadores, A interface hardware/software, 2ª edição, editora UTC.

PMI (2008) A Guide to the Project Management Body of Knowledge (PMBOK Guide). Third Edition ed. [S.l.]: Project Management Institute (PMI). ISBN 1-930699-45-X.

Pressman, Roger S. (2009) Engenharia de Software; 6ª Edição; Editora McGraw-Hill Brasil Técnicos.

Richardson, Robert Jarryl. (1999) Pesquisa social: métodos e técnicas. 3. ed. São Paulo: Atlas, P. 191.

Santos, Antônio Raimundo dos (1999) Metodologia científica: a construção do conhecimento. Rio de Janeiro: DP & A, p. 29.

Seltiz, C. et al. (1967) Métodos de pesquisa nas relações sociais. 2. ed. São Paulo: Ed.da USP.

The Standish Group (2008) "Chaos Report", http://www.standishgroup.com. Última visita: Outubro 2012.

Willrich, Roberto (2004) Sistemas Multimídia Distribuídos. Florianópolis, p. 14, UFSC.

Anexo I – Documento de visão

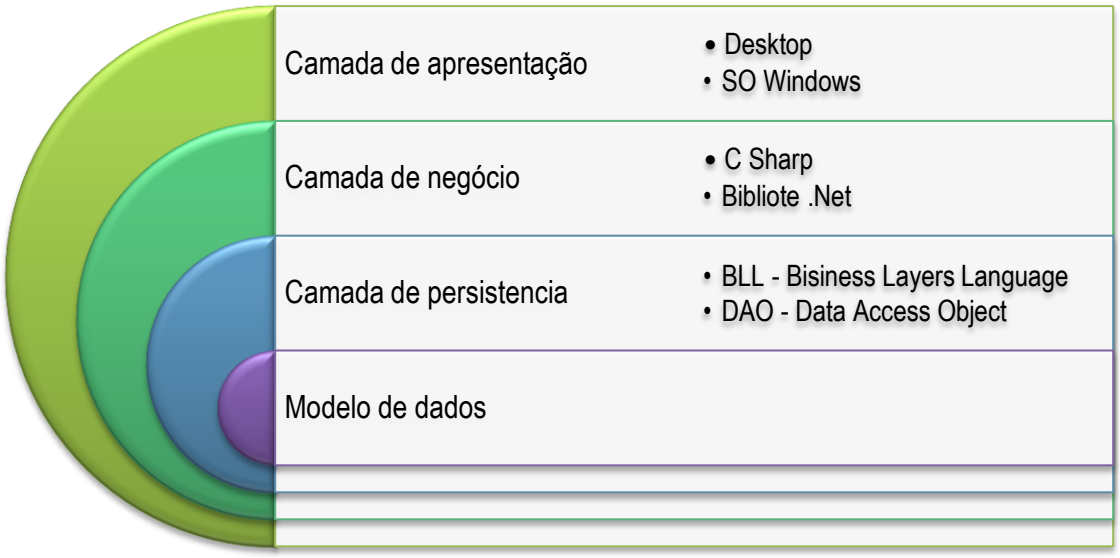
I. Objetivo deste Documento

Este documento apresenta as principais características e lista os requisitos de alto-nível do sistema de gestão de projetos de software. Estas informações foram acordadas entre os principais patrocinadores do projeto e assume-se que serão de conhecimento de qualquer participante do mesmo.

II. Características do Sistema

O sistema será desenvolvido em plataforma desktop utilizando a linguagem c sharp com a biblioteca .Net 4.0 no ambiente de desenvolvimento Visual Dekstop Development. Sua base de dados será gerenciada pelo sistema SQL Server 2008 R2 e seus requisitos serão estabelecidos pelo analista de sistemas deste projeto.

II.a) Interações e Integrações



II.b) Funcionalidades

RF001 – Manter Colaborador – Necessário

Esta funcionalidade compreende a manutenção de colaboradores cadastrados no sistema de gestão de projetos de software, bem como seus dados de login e senha, identificação e contatos e

principalmente seus dados de experiência profissional como escolaridade, cursos e treinamentos, e experiências na função a exercer nos próximos projetos. O sistema permitirá ao gerente listar os colaboradores cadastrados, buscar colaboradores com a opção “Consultar”, alterar dados de colaboradores já cadastrados e cadastrar novos colaboradores com a opção “Salvar”. Esta funcionalidade é necessária, pois permitirá que os colaboradores mantidos sejam incluídos nos projetos a serem executados e administrados por este sistema.

RF002 – Manter Projetos – Necessário

Esta funcionalidade compreende a manutenção de projetos cadastrados no sistema de gestão de projetos de software, bem como as suas datas de cadastro, data prevista para finalização, data de início, cálculo dos custos e recursos alocados para seu desenvolvimento. O sistema permitirá ao gerente de projetos cadastrar todas as informações referentes ao projeto, bem como quantidade de pontos de função, alocar os recursos para o seu desenvolvimento, visualizar prazos estimados para a entrega e principalmente estimar o custo necessário para o desenvolvimento do mesmo. Esta funcionalidade compartilha os dados inseridos no RF001 – Manter Colaborador em virtude de que as atividades necessárias para o desenvolvimento do projeto estão associadas com os colaboradores que irão executá-las.

RF003 – Atividades do colaborador – Necessário

Esta funcionalidade compreende a manutenção das atividades a serem desenvolvidas por cada colaborador referente aos projetos em que os mesmos estão inseridos. O sistema permite ao utilizador visualizar os detalhes da atividade a ser desenvolvida, iniciá-la e posteriormente finalizá-la, listando as mesmas na ordem de execução e principalmente exibindo os prazos para a execução das mesmas, podendo as atividades ultrapassar os prazos estabelecidos, o que implica na funcionalidade de controle de projeto, onde o gerente acompanha e ajusta os prazos do projeto em execução.

RF004 – Controle do projeto – Necessário

Esta funcionalidade é utilizada para captar clientes compradores para realizarem troca ou negociação de um novo veículo com a empresa. O gerente poderá criar uma campanha através de consulta à base de clientes compradores ativos, por meio de filtros pré-determinados, a fim de selecionar alguns para participarem da campanha. O sistema permitirá que o gerente modifique o script desta campanha em uma área de “Configuração” exclusivamente deste trabalho. Nesta área, o gerente também define quais os colaboradores que terão acesso a esta campanha e posteriormente acompanhar os respetivos desempenhos dos mesmos em uma área de “Desempenho” por período. O gerente poderá também atribuir ou não uma data limite para esta campanha. Após a seleção de compradores e selecionados os colaboradores, o sistema disponibilizará os clientes em uma lista que será acessada pelo consultor à medida que o mesmo estiver logado e acessando o sistema de campanha para compradores. Após

selecionar a campanha de clientes compradores, o sistema exibirá os dados do cliente e um script padrão de atendimento podendo “Iniciar” o atendimento aos clientes e/ou “Sair” desta tela. O consultor poderá “registrar um histórico da ligação” e “atribuir um estado à ligação”. Após trabalhar o cliente o sistema exibirá o próximo cliente da lista para ser trabalhado até que não haja mais nenhum cliente atribuído ao consultor.

RF005 – Controle de clientes – Necessário

Esta funcionalidade compreende a manutenção dos clientes (stackholders) solicitantes dos projetos cadastrados para desenvolvimento. O sistema permite ao utilizador visualizar os detalhes dos cadastros dos clientes através de uma listagem geral, bem como altera-lo e excluí-lo.

II.c) Restrições do Produto

Este produto é restrito a ambiente com sistema operacional Windows das versões XP, Vista, 7 e 8.

III. Utilizadores do Sistema

Este sistema é direcionado para utilizadores gestores de projetos.

III.a) Caracterização dos Utilizadores

Gestores de projetos de software que necessitam analisar e acompanhar as atividades inerentes aos seus projetos e auxiliar em suas tomadas de decisões

III.b) Utilização do Sistema

Neste sistema será abordado o acompanhamento de atividades referente ao projeto em desenvolvimento e a disponibilidade dos recursos para novos projetos cadastrados e o custo e prazo dos mesmos diante dos recurso inseridos.

Anexo II – Diagrama de caso de uso

Figura - Diagrama geral de caso de uso.

Anexo III – Documento de especificação de Caso de Uso Manter Cliente

Projeto Sistema de Gestão de Projeto de Software – SGPS 001
Especificação de Caso de Uso:
UC – D2012.001 – Manter Cliente
Versão 1.0.0

Histórico de Revisões

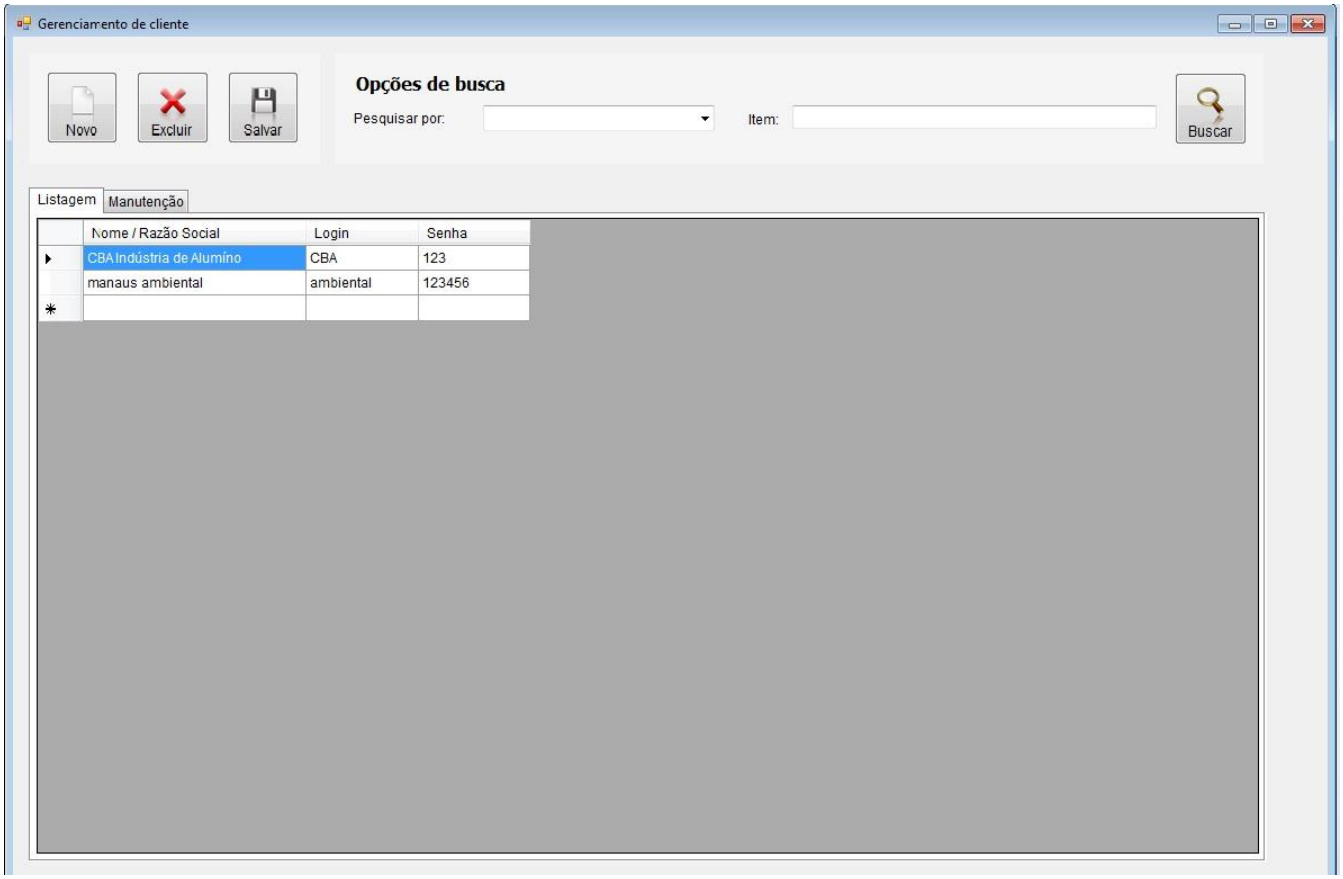
Data	Versão	Descrição	Autor
20/07/2012	001	Especificação de Caso de Uso	Marcelo Silva

1 Breve Descrição

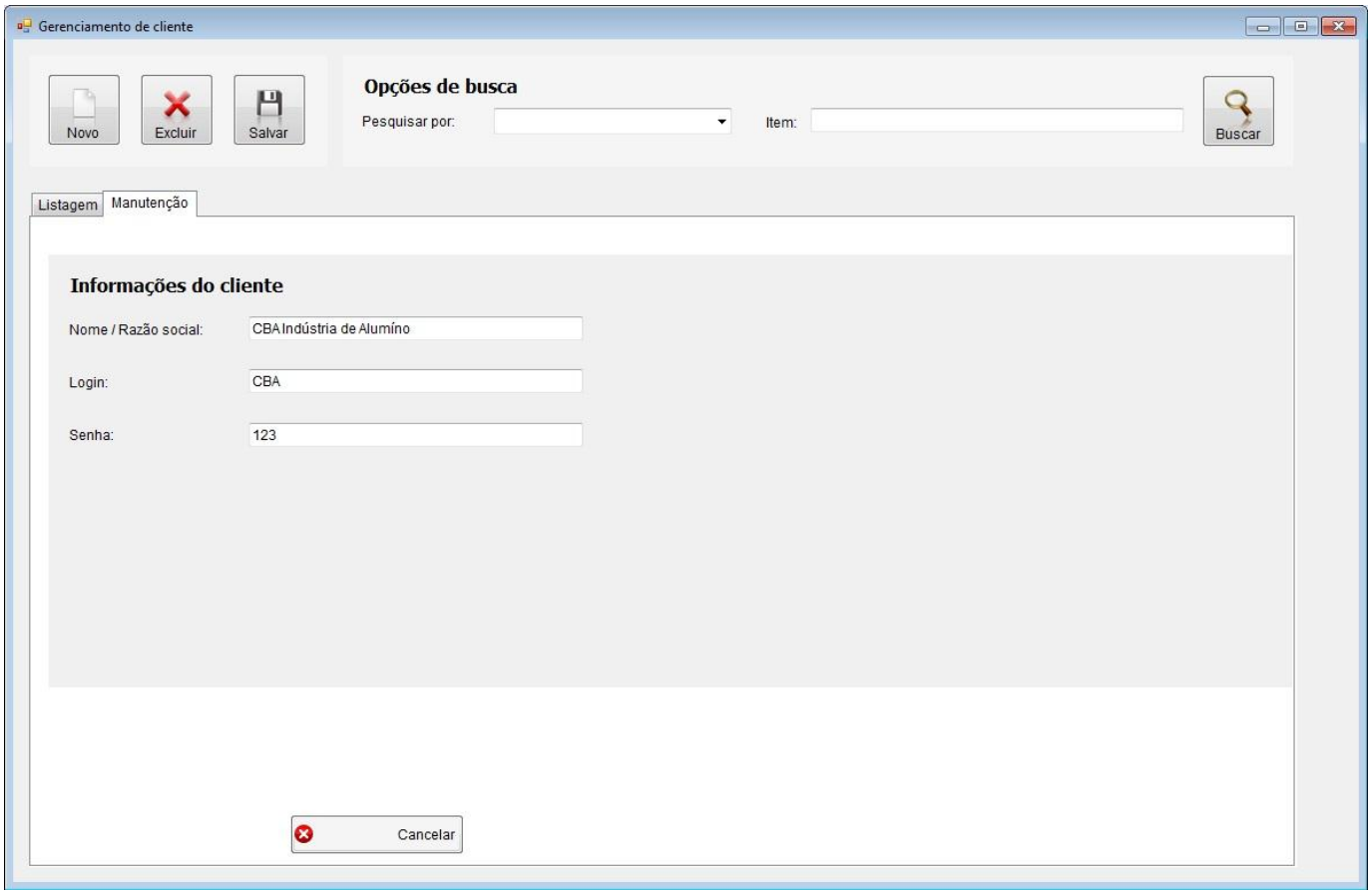
Este caso de uso é responsável por possibilitar ao utilizador a manutenção dos dados do cliente no sistema de gestão de projetos de software.

2 Interfaces – Manter cliente.

2.1 Layout Sugerido



Tela A — Lista de clientes cadastrados



Tela B – Cadastro de ciente

2.2 Relacionamentos com outros casos de uso

Este caso de uso não possui relacionamentos com outros casos de uso.

2.3 Campos

Nº	Nome	Descrição	Valores válidos	Formato	Tipo	Restrições
Tela A – Lista de clientes cadastrados						
1.	Nome / Razão social	Exibe o nome ou razão social do cliente	Até 50 caracteres alfanuméricos	GridView	Texto	Obrigatório Não Alterável
2.	Login	Exibe o login de acesso do cliente	Até 50 caracteres alfanuméricos	GridView	Texto	Obrigatório Não Alterável
3.	Senha	Exibe a senha de acesso do cliente	Até 10 caracteres alfanuméricos	GridView	Texto	Obrigatório Não Alterável
Tela B – Cadastro de ciente						
4.	Nome / Razão social	Especifica o nome ou razão social do cliente	Até 50 caracteres alfanuméricos	TextBox	Texto	Obrigatório Alterável
5.	Login	Especifica o login de acesso do cliente	Até 50 caracteres alfanuméricos	TextBox	Texto	Obrigatório Alterável

6.	Senha	Especifica a senha de acesso do cliente	Até 10 caracteres alfanuméricos	TextBox	Texto	Obrigatório Alterável

2.4 Comandos

Nº	Nome	Ação	Restrições
1	Novo	O sistema exibe a tela B para execução de novo cadastro de cliente	Sempre habilitado
2	Excluir	Exclui o cliente selecionado na grid de cadastrado	Sempre habilitado
3	Salvar	Armazena os dados inseridos do cliente especificado	Sempre habilitado
4	Cancelar	Retorna a página de listagem e não persiste os dados inserido do cliente	Sempre habilitado

3 Realização do Caso de Uso

3.1 Atores

Utilizador (Gestor de projeto de software)

3.2 Pré-condições

O utilizador deve estar CADASTRADO como administrador com o estado ATIVO, e ter acesso ao módulo de gerenciamento de clientes.

3.3 Pós-condições

Após o cadastro de cliente, o mesmo pode acessar as informações do seu projeto a ser desenvolvido.

3.4 Regra de Negócio

- **RN0001 – Manter cliente.**
  - Ao visualizar as campanhas ativas, o sistema exibe os indicadores da campanha na grid.

3.5 Descrição do caso de uso

Fluxo principal – Manter cliente

1. Este caso de uso inicia quando o utilizador (Gestor de projeto de software) acessa o Menu >> Gerenciamento >> Cliente.
2. O sistema exibe a tela A listando os clientes da empresa que possuem cadastro de acesso ao sistema de gestão de projeto de software.
3. Opcionalmente, o gestor poderá os dados de acesso do cliente no sistema.
4. Caso o utilizador queira cadastrar um novo cliente, o mesmo seleciona a opção “Novo”.
5. O sistema exibe a tela B de cadastro de novos clientes.
6. O utilizador especifica as informações de cadastro e seleciona a opção “Salvar”.
7. O sistema armazena os dados especificados e retorna para a tela A.
8. Opcionalmente o utilizador poderá cancelar o cadastro iniciado selecionando a opção “Cancelar”.
9. Este caso de uso se encerra.

Anexo IV – Documento de especificação de Caso de Uso Manter Colaboradores

Projeto Sistema de Gestão de Projeto de Software – SGPS 001
Especificação de Caso de Uso:
UC – D2012.002 – Manter Colaboradores
Versão 1.0.0

Histórico de Revisões

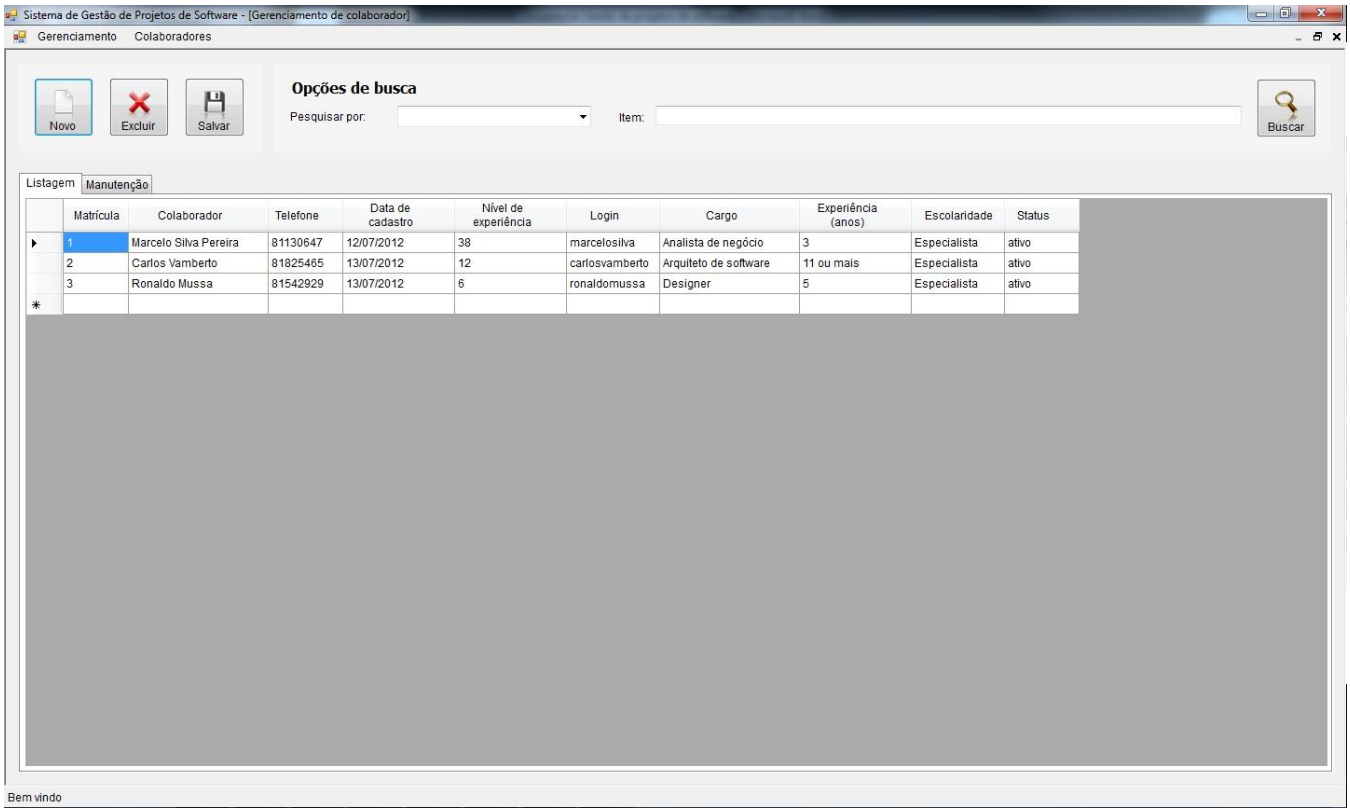
Data	Versão	Descrição	Autor
20/07/2012	001	Especificação de Caso de Uso	Marcelo Silva

1 Breve Descrição

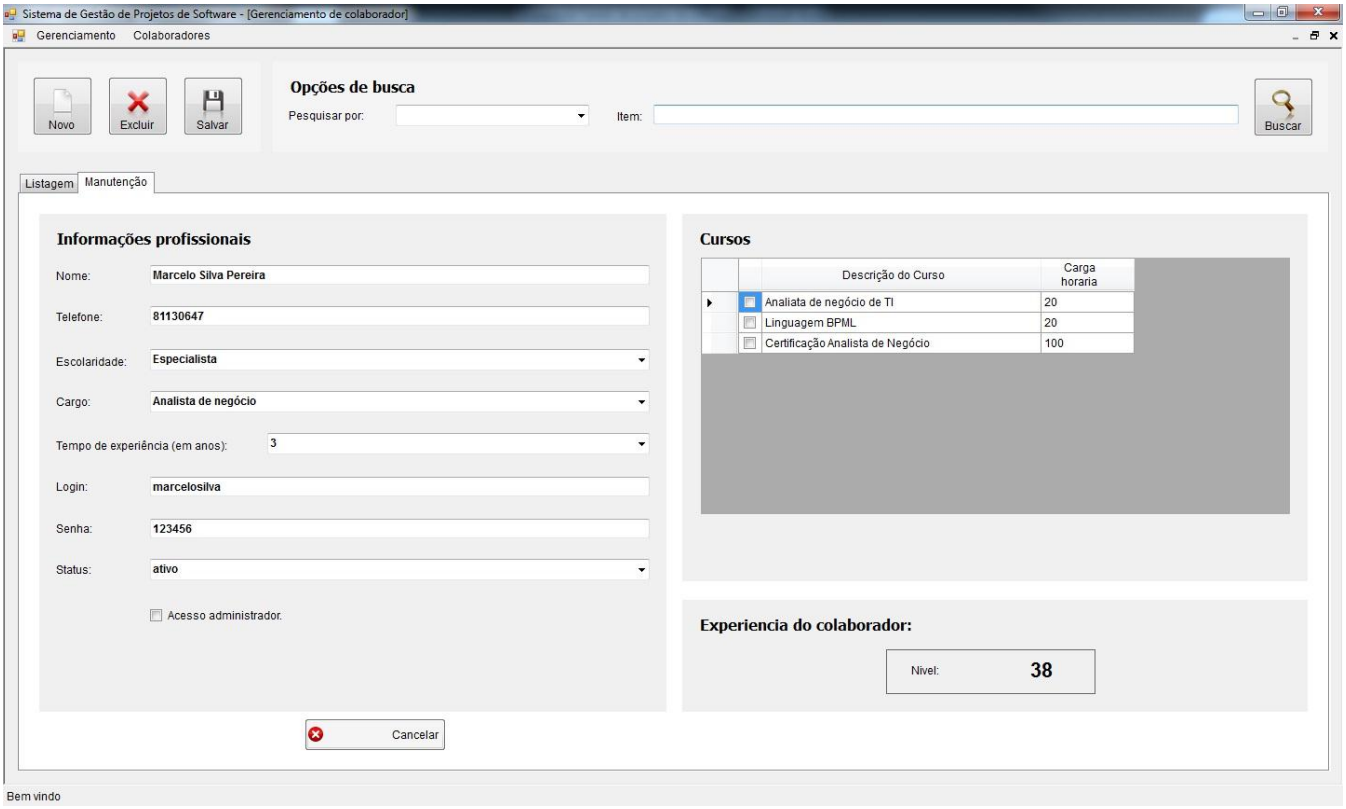
Este caso de uso é responsável por possibilitar ao utilizador a manutenção dos dados dos colaboradores que contribuirão com os projetos a serem gerenciados no sistema de gestão de projetos de software.

2 Interfaces – Manter colaboradores.

2.1 Layout Sugerido



Tela A — Lista de colaboradores



Tela B – Cadastro de colaboradores

2.2 Relacionamentos com outros casos de uso

Este caso de uso não possui relacionamentos com outros casos de uso.

2.3 Campos

Nº	Nome	Descrição	Valores válidos	Formato	Tipo	Restrições
Tela A – Lista de colaboradores						
1.	Matrícula	Exibe o número de matrícula do colaborador	Até 5 caracteres numéricos	GridView	Texto	Obrigatório Não Alterável
2.	Colaborador	Exibe o nome do colaborador cadastrado	Até 50 caracteres alfanuméricos	GridView	Texto	Obrigatório Não Alterável
3.	Telefone	Exibe o número do telefone de contato do cliente	10 caracteres numéricos	GridView	Texto	Obrigatório Não Alterável
4.	Data cadatsro	Exibe a data de cadastro do colaborador	Até 50 caracteres alfanuméricos	GridView	Texto	Obrigatório Não Alterável
5.	Nível de experiencia	Exibe o nível de experiência do colaborador	Até 50 caracteres alfanuméricos	GridView	Texto	Obrigatório Não Alterável
6.	Login	Exibe o login de acesso do colaborador	Até 50 caracteres alfanuméricos	GridView	Texto	Obrigatório Não Alterável
7.	Cargo	Exibe o cargo do colaborador	Até 50 caracteres alfanuméricos	GridView	Texto	Obrigatório Não Alterável
8.	Experiencia	Exibe o nível de experiência do colaborador	De 1 a 11 anos ou mais	GridView	Texto	Obrigatório Não Alterável
9.	Escolaridade	Exibe a escolaridade do colaborador	Até 50 caracteres alfanuméricos	GridView	Texto	Obrigatório Não Alterável
10.	Estado	Exibe o estado de cadastro do colaborador	- Ativo - Inativo	GridView	Texto	Obrigatório Não Alterável
Tela B – Cadastro de ciente						
11.	Nome	Especifica o nome do colaborador a cadastrar	Até 50 caracteres alfanuméricos	TextBox	Texto	Obrigatório Alterável
12.	Telefone	Especifica o login de acesso do cliente	Até 50 caracteres alfanuméricos	TextBox	Texto	Não Obrigatório Alterável
13.	Escolaridade	Lista as opções de escolaridade do colaborador	Até 50 caracteres alfanuméricos	ComboBox	Texto	Obrigatório Alterável
14.	Cargo	Lista as opções de cargo do colaborador	Até 50 caracteres alfanuméricos	ComboBox	Texto	Obrigatório Alterável
15.	Tempo de experiencia (em anos)	Lista as opções de tempo de experiência do colaborador	De 1 a 11 anos ou mais	ComboBox	Texto	Obrigatório Alterável

16.	Login	Especifica o login do colaborador	Até 50 caracteres alfanuméricos	TextBox	Texto	Não Obrigatório Alterável
17.	Senha	Especifica a senha do colaborador	Até 10 caracteres alfanuméricos	TextBox	Texto	Não Obrigatório Alterável
18.	Estado	Lista os estado possíveis do colaborador	- Ativo - Inativo	ComboBox	Texto	Obrigatório Alterável
19.	Experiencia do colaborador	Exibe o nível de experiência do colaborador	Até 3 caracteres numéricos	Label	Texto	Obrigatório Não Alterável

2.4 Comandos

Nº	Nome	Ação	Restrições
1	Novo	O sistema exibe a tela B para execução de novo cadastro de colaborador.	Sempre habilitado
2	Excluir	Exclui o colaborador selecionado na grid de cadastrado	Sempre habilitado
3	Salvar	Armazena os dados inseridos do colaborador especificado	Sempre habilitado
4	Cancelar	Retorna a página de listagem e não persistem os dados inseridos do colaborador	Sempre habilitado

3 Realização do Caso de Uso

3.1 Atores

Utilizador (Gestor de projeto de software)

3.2 Pré-condições

O utilizador deve estar CADASTRADO como administrador com o estado ATIVO, e ter acesso ao módulo de gerenciamento de colaboradores.

3.3 Pós-condições

Após o cadastro de colaborador, o mesmo fica disponível para ser inserido em projetos e posteriormente visualizar suas atividades a realizar.

3.4 Regra de Negócio

- **RN0001 – Manter colaborador.**
  - Não se aplica a esse caso de uso.

3.5 Descrição do caso de uso

Fluxo principal – Manter colaborador

1. Este caso de uso inicia quando o utilizador (Gestor de projeto de software) acessa o Menu >> Gerenciamento >> Colaborador.
2. O sistema exibe a tela A listando os colaboradores que possuem cadastro no sistema de gestão de projeto de software.

3. Opcionalmente, o gestor poderá alterar os dados de cadastro do colaborador no sistema.
4. Caso o utilizador queira cadastrar um novo colaborador, o mesmo seleciona a opção “Novo”.
5. O sistema exibe a tela B de cadastro de novos colaboradores.
6. O utilizador especifica as informações de cadastro e seleciona a opção “Salvar”.
7. O sistema armazena os dados especificados e retorna para a tela A.
8. Opcionalmente o utilizador poderá cancelar o cadastro iniciado selecionando a opção “Cancelar”.
9. Este caso de uso se encerra.

Anexo V – Documento de especificação de Caso de Uso Manter Empresa

Projeto Sistema de Gestão de Projeto de Software – SGPS 001
Especificação de Caso de Uso:
UC – D2012.003 – Manter Empresa
Versão 1.0.0

Histórico de Revisões

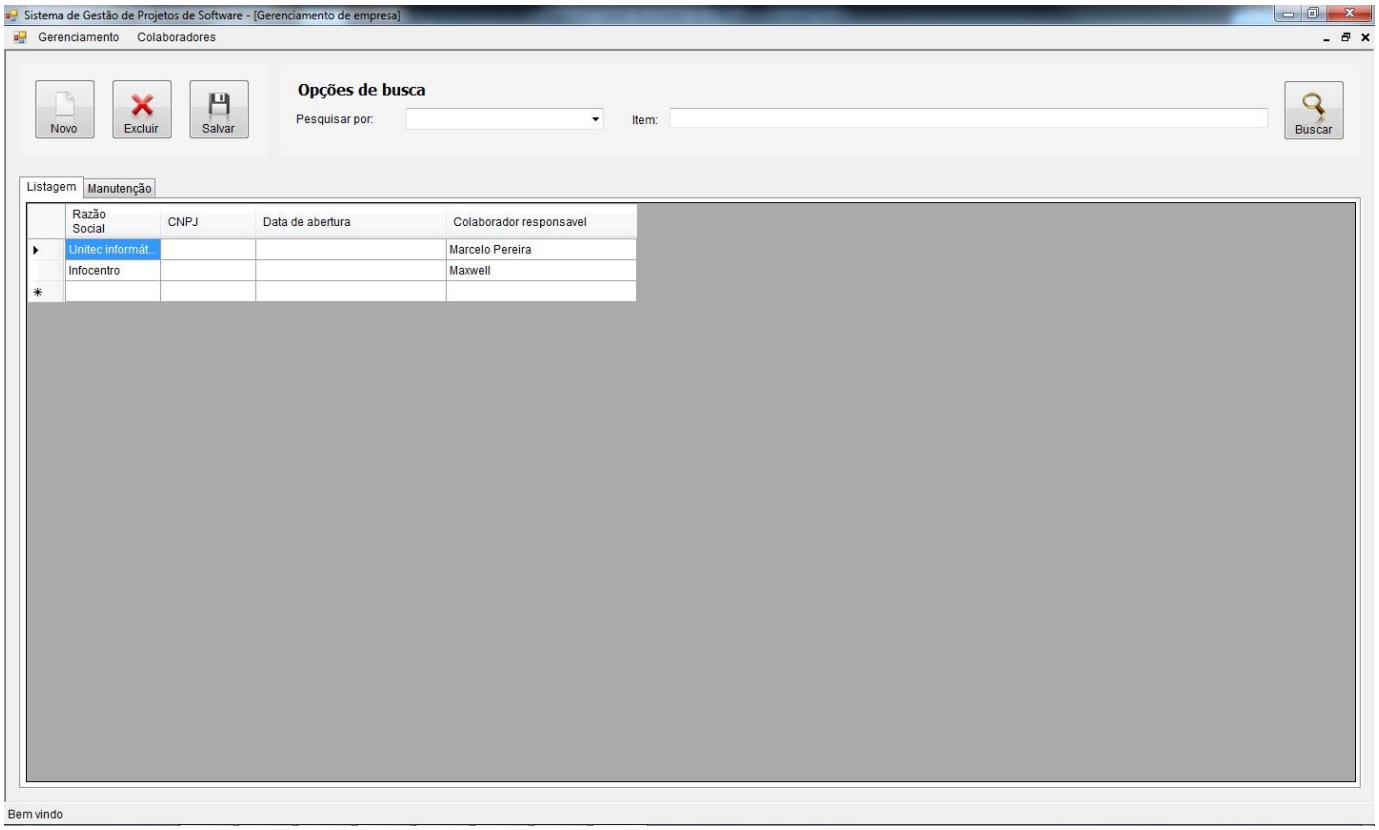
Data	Versão	Descrição	Autor
20/07/2012	001	Especificação de Caso de Uso	Marcelo Silva

1 Breve Descrição

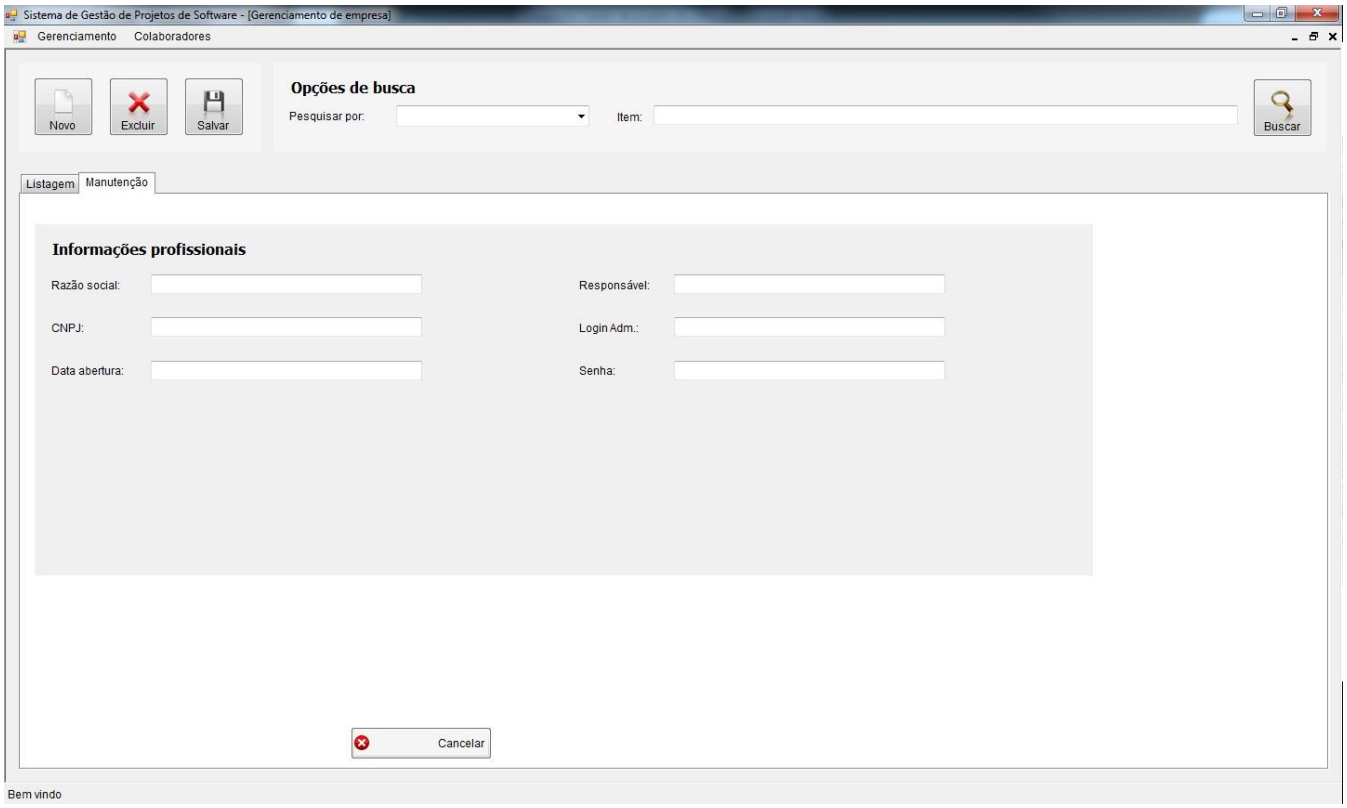
Este caso de uso é responsável por possibilitar ao utilizador a manutenção dos dados da empresa que irá administrar o sistema e gerenciar o projeto de software a ser gerenciado no sistema de gestão de projetos de software.

2 Interfaces – Manter empresa.

2.1 Layout Sugerido



Tela A – Lista de empresas



Tela B – Cadastro de empresas

2.2 Relacionamentos com outros casos de uso

Este caso de uso não possui relacionamentos com outros casos de uso.

2.3 Campos

Nº	Nome	Descrição	Valores válidos	Formato	Tipo	Restrições
Tela A – Lista de colaboradores						
1.	Razão social	Exibe a razão social da empresa cadastrada	Até 50 caracteres alfanuméricos	GridView	Texto	Obrigatório Não Alterável
2.	CNPJ	Exibe o CNPJ da empresa cadastrada	Até 50 caracteres alfanuméricos	GridView	Texto	Não Obrigatório Não Alterável
3.	Data abertura	Exibe a data de abertura da empresa	10 caracteres alfanuméricos no formato 00/00/0000	GridView	Texto	Não Obrigatório Não Alterável
4.	Colaborador responsável	Exibe o nome do colaborador responsável	Até 50 caracteres alfanuméricos	GridView	Texto	Obrigatório Não Alterável
Tela B – Cadastro de empresas						
5.	Razão social	Especifica o nome da empresa responsável	Até 50 caracteres alfanuméricos	TextBox	Texto	Obrigatório Alterável
6.	CNPJ	Especifica o CNPJ da empresa responsável	Até 50 caracteres alfanuméricos	TextBox	Texto	Não Obrigatório Alterável
7.	Data abertura	Especifica a data de abertura da empresa responsável	10 caracteres alfanuméricos no formato 00/00/0000	TextBox	Texto	Não Obrigatório Alterável
8.	Colaborador responsável	Especifica o nome do colaborador responsável pela empresa	Até 50 caracteres alfanuméricos	TextBox	Texto	Obrigatório Alterável
9.	Login admin	Especifica o login de administrador do sistema	Até 50 caracteres alfanuméricos	TextBox	Texto	Obrigatório Alterável
10.	Senha	Especifica a senha de acesso do responsável empresa	Até 50 caracteres alfanuméricos	TextBox	Texto	Obrigatório Alterável

2.4 Comandos

Nº	Nome	Ação	Restrições
1	Novo	O sistema exibe a tela B para execução de novo cadastro de empresa.	Sempre habilitado
2	Excluir	Exclui a empresa selecionada na grid de cadastrado	Sempre habilitado
3	Salvar	Armazena os dados inseridos da empresa especificada	Sempre habilitado
4	Cancelar	Retorna a página de listagem e não persistem os dados inseridos da empresa	Sempre habilitado

3 Realização do Caso de Uso

3.1 Atores

Utilizador (Gestor de projeto de software)

3.2 Pré-condições

O utilizador deve estar CADASTRADO como administrador com o estado ATIVO, e ter acesso ao módulo de gerenciamento de empresas.

3.3 Pós-condições

Após o cadastro de empresas, o mesmo fica disponível para cadastrar seus colaboradores e clientes, consequentemente projetos a serem desenvolvidos.

3.4 Regra de Negócio

- **RN0001 – Manter empresa.**
  - Não se aplica a esse caso de uso.

3.5 Descrição do caso de uso

Fluxo principal – Manter empresa

1. Este caso de uso inicia quando o utilizador (Gestor de projeto de software) acessa o Menu >> Gerenciamento >> Empresa.
2. O sistema exibe a tela A listando as empresas que possuem cadastro no sistema de gestão de projeto de software.
3. Opcionalmente, o gestor poderá alterar os dados de cadastro da empresa no sistema.
4. Caso o utilizador queira cadastrar uma nova empresa, o mesmo seleciona a opção “Novo”.
5. O sistema exibe a tela B de cadastro de novas empresas.
6. O utilizador especifica as informações de cadastro e seleciona a opção “Salvar”.
7. O sistema armazena os dados especificados e retorna para a tela A.
8. Opcionalmente o utilizador poderá cancelar o cadastro iniciado selecionando a opção “Cancelar”.
9. Este caso de uso se encerra.

Anexo VI – Documento de especificação de Caso de Uso Manter Projeto

Projeto Sistema de Gestão de Projeto de Software – SGPS 001
Especificação de Caso de Uso:
UC – D2012.004 – Manter Projeto
Versão 1.0.0

Histórico de Revisões

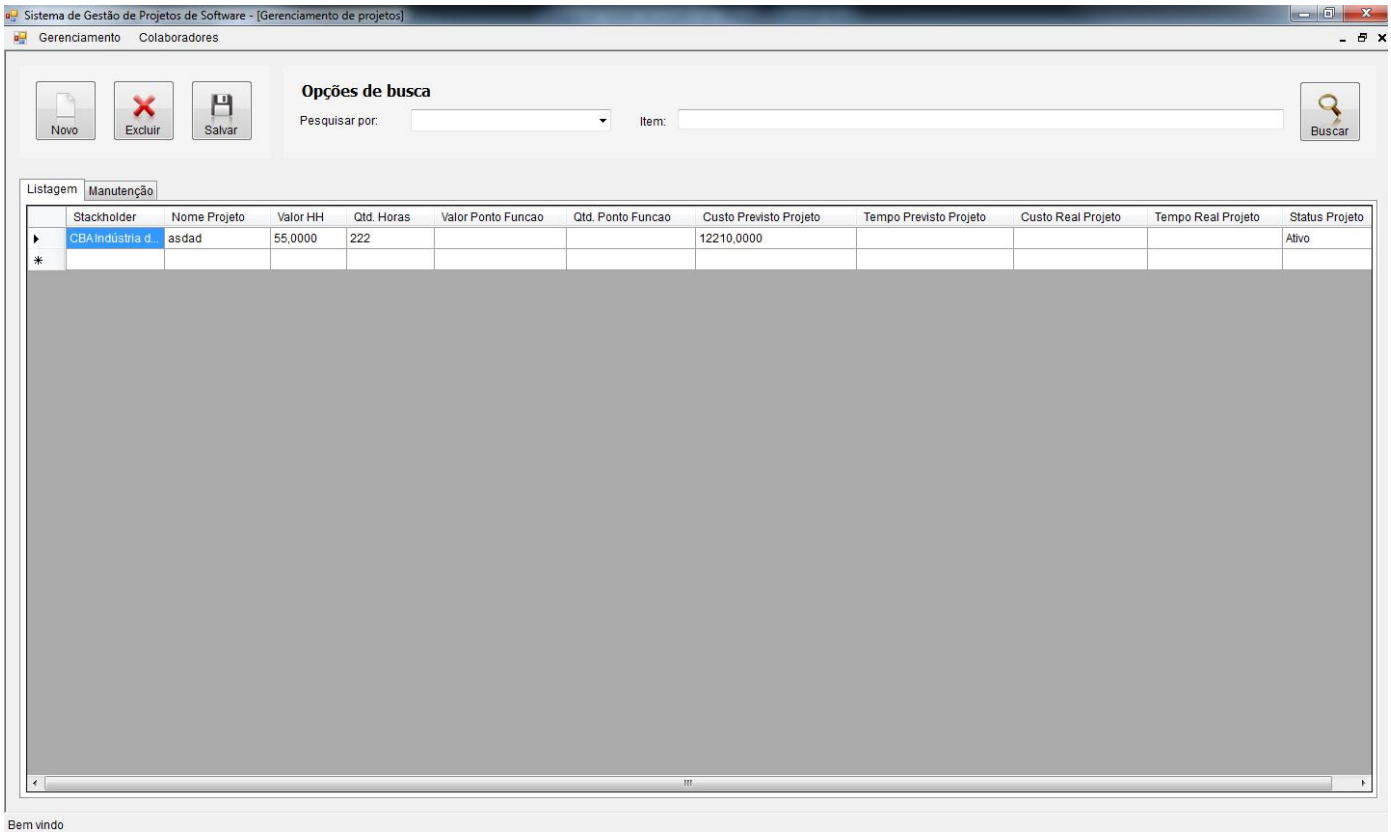
Data	Versão	Descrição	Autor
01/08/2012	001	Especificação de Caso de Uso	Marcelo Silva

1 Breve Descrição

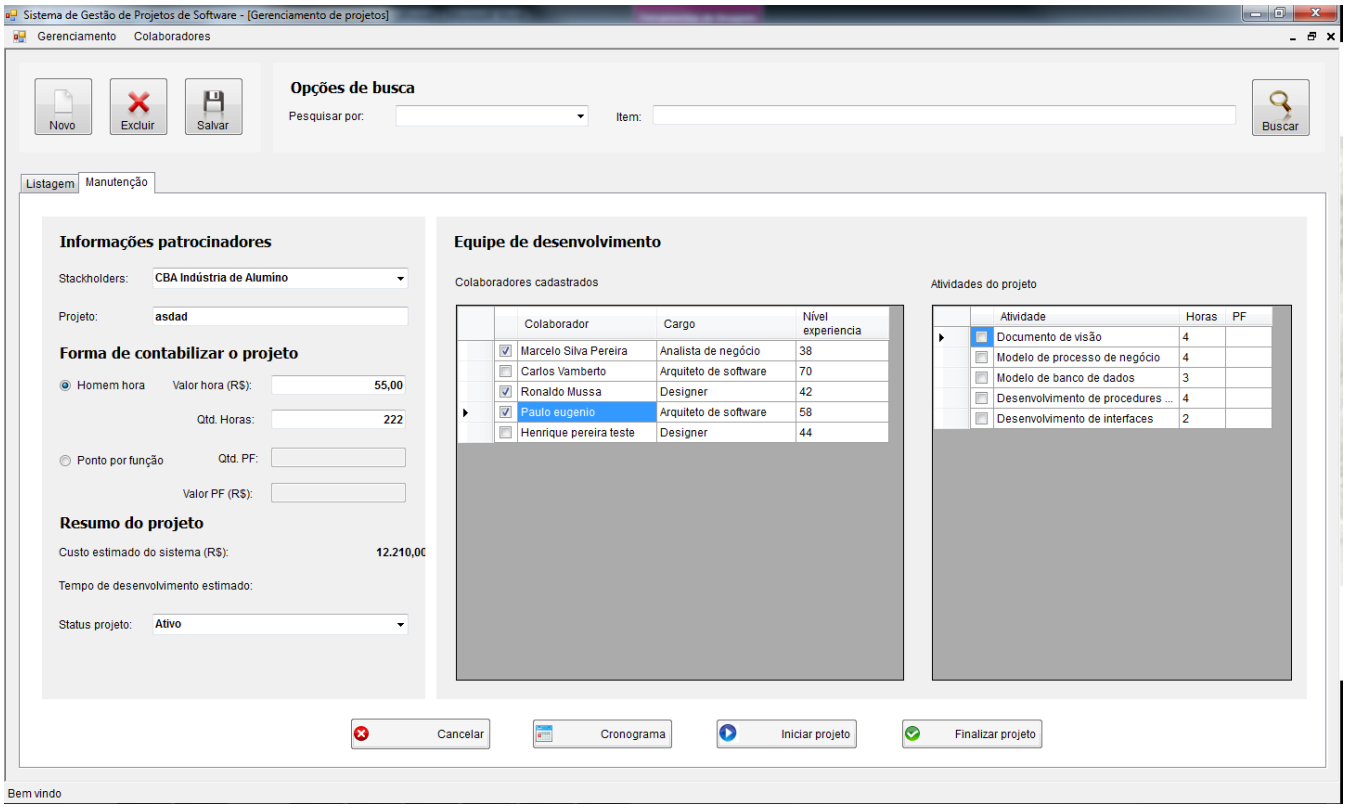
Este caso de uso é responsável por possibilitar ao utilizador a manutenção dos dados a serem desenvolvidos pela empresa no sistema de gestão de projetos de software.

2 Interfaces – Manter projeto.

2.1 Layout sugerido



Tela A – Lista de projetos



Tela B – Cadastro de projetos

2.2 Relacionamentos com outros casos de uso

Este caso de uso possui relacionamentos com o caso de uso UC-D2012.001 Manter Cliente, UC-D2012.002 Manter Colaborador e UC-D2012.005 Manter Atividade.

2.3 Campos

Nº	Nome	Descrição	Valores válidos	Formato	Tipo	Restrições
Tela A – Lista de projetos						
1.	Stackholder	Exibe o nome da empresa solicitante do projeto	Até 50 caracteres alfanuméricos	GridView	Texto	Obrigatório Não Alterável
2.	Nome projeto	Exibe o nome do projeto a ser desenvolvido	Até 50 caracteres alfanuméricos	GridView	Texto	Obrigatório Não Alterável
3.	Valor HH	Exibe o valor a ser cobrado pela hora ao projeto	Até 10 caracteres alfanuméricos no formato **000.000,00**	GridView	Texto	Não Obrigatório Não Alterável
4.	Qtd. Horas	Exibe a quantidade de horas que será desprendida ao projeto		GridView	Texto	Não Obrigatório Não Alterável
5.	Valor ponto função			GridView	Texto	Não Obrigatório Não Alterável
6.	Qtd. ponto função			GridView	Texto	Não Obrigatório Não Alterável
7.	Custo previsto projeto	Exibe o nome do colaborador responsável	Até 50 caracteres alfanuméricos	GridView	Texto	Obrigatório Não Alterável
8.	Tempo previsto projeto			GridView	Texto	Obrigatório Não Alterável
9.	Custo real projeto			GridView	Texto	Obrigatório Não Alterável
10.	Tempo real projeto			GridView	Texto	Obrigatório Não Alterável
11.	Estado projeto			GridView	Texto	Obrigatório Não Alterável
Tela B – Cadastro de empresas						
12.	Razão social	Especifica o nome da empresa responsável	Até 50 caracteres alfanuméricos	TextBox	Texto	Obrigatório Alterável

13.	CNPJ	Especifica o CNPJ da empresa responsável	Até 50 caracteres alfanuméricos	TextBox	Texto	Não Obrigatório Alterável
14.	Data abertura	Especifica a data de abertura da empresa responsável	10 caracteres alfanuméricos no formato 00/00/0000	TextBox	Texto	Não Obrigatório Alterável
15.	Colaborador responsável	Especifica o nome do colaborador responsável pela empresa	Até 50 caracteres alfanuméricos	TextBox	Texto	Obrigatório Alterável
16.	Login admin	Especifica o login de administrador do sistema	Até 50 caracteres alfanuméricos	TextBox	Texto	Obrigatório Alterável
17.	Senha	Especifica a senha de acesso do responsável empresa	Até 50 caracteres alfanuméricos	TextBox	Texto	Obrigatório Alterável

2.4 Comandos

Nº	Nome	Ação	Restrições
5	Novo	O sistema exibe a tela B para execução de novo cadastro de empresa.	Sempre habilitado
6	Excluir	Exclui a empresa selecionada na grid de cadastrado	Sempre habilitado
7	Salvar	Armazena os dados inseridos da empresa especificada	Sempre habilitado
8	Cancelar	Retorna a página de listagem e não persistem os dados inseridos da empresa	Sempre habilitado

3 Realização do Caso de Uso

3.6 Atores

Utilizador (Gestor de projeto de software)

3.7 Pré-condições

O utilizador deve estar CADASTRADO como administrador com o estado ATIVO, e ter acesso ao módulo de gerenciamento de empresas.

3.8 Pós-condições

Após o cadastro de empresas, o mesmo fica disponível para cadastrar seus colaboradores e clientes, consequentemente projetos a serem desenvolvidos.

3.9 Regra de Negócio

- **RN0001 – Manter empresa.**
  - Não se aplica a esse caso de uso.

3.10 Descrição do caso de uso

Fluxo principal – Manter empresa

10. Este caso de uso inicia quando o utilizador (Gestor de projeto de software) acessa o Menu >> Gerenciamento >> Empresa.
11. O sistema exibe a tela A listando as empresas que possuem cadastro no sistema de gestão de projeto de software.
12. Opcionalmente, o gestor poderá alterar os dados de cadastro da empresa no sistema.
13. Caso o utilizador queira cadastrar uma nova empresa, o mesmo seleciona a opção “Novo”.
14. O sistema exibe a tela B de cadastro de novas empresas.
15. O utilizador especifica as informações de cadastro e seleciona a opção “Salvar”.
16. O sistema armazena os dados especificados e retorna para a tela A.
17. Opcionalmente o utilizador poderá cancelar o cadastro iniciado selecionando a opção “Cancelar”.
18. Este caso de uso se encerra.