INSTITUTO SUPERIOR TÉCNICO
Universidade Técnica de Lisboa

O Processo de Manutenção de Software num Contexto de Full-Outsourcing

Maria do Carmo Silva Santos Cardim

Dissertação para obtenção do Grau de Mestre em

Engenharia Informática e de Computadores

Júri

Presidente: Prof. Pedro Manuel Moreira Vaz Antunes de Sousa

Orientador: Prof. Miguel Leitão Bignolas Mira da Silva

Setembro de 2007

Resumo

O resultado do desenvolvimento de software é a entrega de um produto que satisfaça os requisitos do utilizador. Posto isto, o produto tem de mudar e evoluir. Uma vez operacional e entregue são descobertas anomalias, e novos requisitos emergem pelo lado dos utilizadores. Começa assim uma nova fase do ciclo de vida do software: a manutenção. É nesta fase que 70% a 80% do esforço total do ciclo de vida do software é empenhado, visto abranger um grande número de recursos humanos que desempenham um trabalho repetitivo e monótono.

A exposição deste facto na comunidade científica originou um aumento do número de estudos sobre o tema da Manutenção de Software (MS). Esses estudos dividem-se essencialmente em três grandes grupos. O primeiro prende-se com a definição das áreas da MS: Adaptativa, Correctiva, Evolutiva e Preventiva; e com os estudos levam às suas definições. O segundo é relacionado com o esforço envolvido durante a MS, e a sua consequência para as empresas, apresentando o outsourcing como solução para a diminuição desse mesmo esforço. O outsourcing permite uma redução do tempo dispendido aumentando a produtividade efectiva, mas não garante, só por si, a eficiência e a eficácia do processo associado à manutenção. O terceiro prende-se com o facto de não existir um processo considerado standard pelas organizações. Apenas recentemente começaram a emergir boas práticas relacionadas com este tema, nomeadamente o ITIL (Information Technology Infrastructure Library).

O meu estudo centrar-se-á acima de tudo no desenho do processo de MA. A razão desta escolha prende-se com o facto de, após investigação efectuada, o ter considerado, regra geral, ineficaz e ineficiente, necessitando consequentemente de resolução. A solução proposta passa por integrar os conceitos das boas práticas (ITIL) num processo que opere em *outsourcing*, aproveitando as vantagens de ambas as soluções. As vantagens mais evidentes são: a redução de custos para a Manutenção de Software, consequentes da criação de uma economia de escala para as empresas de *outsourcing*.

O desenho do processo de MS, foi feito com base num caso de estudo. O caso de estudo é o processo de Manutenção Evolutiva (ME), de um Departamento de Sistemas de Informação que opera em *full-outsourcing*. Foi utilizada uma ferramenta para comparar os seguintes processos: Processo antigo de ME, processo novo de ME e processo de Gestão de Alterações (ITIL). Através da análise dos resultados pode-se concluir que o ITIL pode ser aplicado em empresas que operam em *full-outsourcing,* e, visto ser o factor variante entre o processo antigo e o novo, tem um impacto notório e positivo sobre o processo afectado. O impacto refere-se ao número de horas que irá ser poupado, tanto a curto como a médio e longo prazo, e num aumento indirecto de produtividade.

Palavras-chave: manutenção de software, alteração, custo, esforço, *outsourcing*, ITIL

Abstract

The result of software development is the delivery of a product that satisfies the user requirements. As such, the product has to change and evolve. Once it is operational and delivered, users discover anomalies and new requirements emerge. Thus, a new phase of the software life cycle starts: the maintenance phase. In this phase, 70% to 80% of the total effort of the software is pledged, since it demands a great number of human resources that play a repetitive and monotonous work.

The exposition of this reality in the scientific community originated an increase in the number of studies concerning Software Maintenance (SM). These studies are essentially divided into three large groups. The first addresses the definition of the SM areas (Adaptive, Perfective, Corrective, and Preventive) which gain stability by the end of the 90's. The second is related to the effort evolved during the SM, and its consequence to the corporations, presenting the outsourcing as a solution to the reduction of that same effort. The outsourcing allows a reduction of time dispended, increasing the effective productivity, but it doesn't guarantee, on its own, the efficiency and efficacy of the process associated to the SM. The third deals with the fact that there is no process considered standard by the organizations. Only recently good practices related to the subject began emerging, namely the ITL (Information Technology Infrastructure Library).

Of the exposed subjects, my study will deal mainly with SM process illustration. The reason for that is that, after investigation, having considerate it, inefficient and inefficacy, therefore needing a resolution. The proposed solution consists of integrating the concept of good practices (ITIL) into a process that operates outsourcing, taking full advantages of both solutions. The most obvious advantages are: cost reduction for the SM, due to the creation of an economy of scales for the outsourcing companies.

The SM process illustration was created based on a case study. The case study is the process of the Perfective Maintenance (PM), of an Information Systems Department which operates in full-outsourcing. A tool was used to compare the following processes: Old PM process, new PM process and the Change Management process (ITIL). The conclusion taken after analyzing the results is that the ITIL can be applied to companies that operate in full-outsourcing and, since it is the variant factor between the old process and the new process, it has a notorious and positive impact on the affected process. The impact refers to the number of hours that will be saved, in the short, medium, and long run and in an indirect increase of the productivity.

Key-words: software maintenance, change, cost, effort, outsourcing, ITIL

Agradecimentos

Desejo consignar aqui a minha gratidão a várias pessoas que contribuíram com diversas ajudas para a realização deste trabalho.

Ao Professor Miguel Mira da Silva, por despertar a minha curiosidade e inspirar a minha investigação no presente tema, oferecendo-me, o seu conhecimento, material de estudo e disponibilidade e paciência ao longo deste ano.

À minha família, aos meus pais, Madalena e Carlos, aos meus irmãos, Maria e Carlos Maria, e à minha cunhada, Inês, que não se pouparam a esforços para que se tornasse realidade, não só este trabalho como também a concretização desta minha etapa pessoal e profissional. Aos meus sobrinhos, João, António e Francisca, pela sua alegria que contagiou, os meus momentos mais difíceis.

Aos meus colegas e amigos, Joana Esteves, Diogo Gonçalves e Luís Soares, que sempre me apoiaram, e nunca me deixaram desistir. Obrigada pelas críticas e pela partilha de conhecimento.

A todos os meus amigos, especialmente à Catarina Orta, pela sua amizade, compreensão e disponibilidade.

Maria do Carmo Cardim

Índice

Lista de Figuras

Lista de Tabelas

Lista de Abreviações

RFC – Request For Change

DSI – Departamento de Sistemas de Informação

MA – Manutenção Aplicacional

MS – Manutenção de Software

ITIL – Information Technology Infrastructure Library

CMDB - Configuration Management Database

1. Introdução

O ciclo de vida do software é composto pelas seguintes fases: requisitos, desenho, construção, testes e manutenção. Neste estudo vamos dar relevância à parte integrante do ciclo de vida, que historicamente tem recebido menos atenção que as outras fases: a manutenção.

Durante muito tempo, a fase que esteve em grande plano na maioria das organizações foi a fase de desenvolvimento. Com o início do desenvolvimento dos Sistemas de Informação (SI), todas as empresas criavam os seus próprios produtos de software, e portanto eram responsáveis por todo o ciclo de vida do software. Contudo, à medida que se vai podendo medir os valores investidos em cada parte do ciclo de vida esta realidade começa a alterar-se. As organizações lutam por melhores resultados do seu investimento em desenvolvimento, mantendo o software operacional durante o maior período de tempo possível, aumentando assim a fase da manutenção. Além disso, a disponibilização de ferramentas e linguagens de desenvolvimento *opensource* atraiu as atenções, visto que é possível manter código desenvolvido por outros, ou vice-versa.

Contudo, a manutenção é uma actividade dispendiosa. A manutenção de software inclui correcção de erros, alterações e melhoramentos ao software operacional. Com utilizadores em acção, existe sempre espaço para melhoramentos técnicos e operacionais no sistema existente.

Por estas razões, existem oportunidades para pesquisar este tema de modo a encontrar meios para aumentar a produtividade das actividades de manutenção.

Uma grande oportunidade está no estudo do ciclo de vida das aplicações, mais propriamente na fase de manutenção das aplicações. Esta tem normalmente o custo de 70% do custo total de uma aplicação desde a sua definição de requisitos, até ao seu fecho. Esse custo é aceitável visto que abrange a maior parte do ciclo de vida da aplicação, mas existe sempre o objectivo de diminuir o mesmo melhorando o processo utilizado para a própria manutenção. A falta de técnicas e experiência em sistemas de informação faz com que as empresas estejam a apostar numa solução em que apenas são responsáveis por controlar e não por implementar: o *outsourcing*.

O *outsourcing* de TI é uma nova solução para o desenvolvimento e gestão de aplicações de TI, contudo a falta de compreensão dos sistemas de gestão de TI, torna esta solução difícil de vencer e de se tornar numa realidade comum.

O problema central da pesquisa, realizada no capítulo 2, é a inabilidade de mudar facilmente e rapidamente o software. Este problema limita as empresas e é a causa predominante do descontentamento do utilizador com os sistemas de informação. O problema levanta-se porque o software é orientado ao produto, independentemente de este ser comprado, alugado ou elaborado por terceiros. Além da configuração relativamente simples, o software no ponto de entrega é monolítico, o que traz muitas características indesejáveis tais como funcionalidade, melhoramentos e dificuldade não desejados na resposta à mudança rápida do negócio (Bennett, et al., 2000).

Existem várias formas de resolver este problema, e essas passam por elaborar um processo para a realização da manutenção de software. Com o aparecimento das melhores práticas (ITIL), finalmente foi dado o ênfase necessário a fase de manutenção, estendendo-se a várias áreas da mesma. Apesar de existirem procedimentos "ideais", a realidade das empresas encontra-se a anos-luz dessas boas práticas. Por isso, temos de delinear um processo/procedimento que se enquadre na realidade da empresa tendo por base as boas práticas. O enquadramento de ambas é necessário visto que as empresas existentes têm neste momento os seus próprios métodos de lidar com a manutenção das aplicações.

A fase de manutenção concentra 70% dos custos totais do ciclo de vida do software. Apesar deste facto, praticamente relatado desde os anos 80, a comunidade de investigação não tem demonstrado uma grande preocupação em encontrar um método adequado para a resolução desta questão, com vista à diminuição dos custos desta fase. Posto isto, as empresas recorrem ao *outsourcing* para a manutenção de software e ainda ao *outsourcing* de TI procurando os melhores SLAs de acordo com os níveis de serviço que pretendem obter. O problema do método de resolução da manutenção de software continua assim por resolver. Contudo os custos têm tendência a diminuir, devido à competitividade das empresas de *Outsourcing* de TI.

Alguns modelos apresentados são baseados em boas práticas como o ITIL e o CMM de software. Em ambos os casos existe um *blueprint* do que deve existir na organização, mas não a forma de implementar. Esta área é claramente um dos desafios actuais das empresas, visto existir a necessidade de descrever como os clientes e utilizadores podem ter acesso aos serviços apropriados de suporte as suas actividades e ao negócio, e a forma como esses serviços são suportados.

1.1. Motivação

A Manutenção de Software ganhou ênfase no ciclo de vida do Software, devido ao elevado esforço envolvido para a sua realização (Foster, 1993). Este esforço é maior do que o esperado, tendo em conta o investimento pretendido na concepção de um novo software.

É imperativo expor os problemas que levam a estes valores bem como testar soluções para equilibrar os mesmos com os valores das outras fases. As soluções podem passar por colocar a Manutenção de Software no contexto de *outsourcing* e destacar esta fase de entre as outras fases do ciclo de vida do software.

A necessidade de contextualizar a manutenção de um software, é um desafio que se prende com a diminuição do esforço envolvido nesta fase, tanto pelo lado do negócio (quem pede as alterações) como pelo lado do programador (que deveria receber tarefas claras e objectivas). Acredito que um processo que integre tanto as melhores práticas bem como as normas e conceitos de uma empresa seja condição necessária para diminuir o esforço envolvido nesta actividade, providenciando assim um ponto de partida para o desenvolvimento ideal da tarefa para que foi criada.

1.2. Organização da Tese

No capítulo 2 está descrito o estado da arte face ao tema da tese: A manutenção de software num contexto de full-*outsourcing*. O estado da arte está dividido em 3 capítulos principais, de modo a introduzir as bases para a apresentação do problema e consequente proposta.

Estes capítulos são: Manutenção de Software, onde estão descritos os conceitos base através das definições encontradas pela comunidade cientifica; Gestão da Manutenção, onde estão descritas duas abordagens de gestão, uma através da biblioteca de boas práticas na qual se baseou a ISO 20000 (ITIL) e a outra análoga ao CMM denominada Modelo de Maturidade de Manutenção de Software; *Outsourcing*, onde se faz a introdução aos conceitos do *outsourcing*, seus benefícios e suas dificuldades.

A apresentação do problema é realizada no capítulo 3. Neste capítulo é feito um resumo de todos os problemas subjacentes ao tema e ainda é identificado um ponto de partida para uma proposta de solução.

No capítulo 4, é apresentada uma proposta de solução – o cruzamento das três áreas descritas no capítulo 2, e ainda são enunciadas as vantagens da implementação de cada uma delas para a resolução do problema.

O capítulo 5 é a exposição do caso de estudo. Esta exposição passa por apresentar a empresa onde está colocado o processo de manutenção em questão, e enquadrar todo o trabalho desenvolvido ao longo do último ano. As tarefas que se destacam no caso de estudo são: o desenho do processo de manutenção e a implementação do protótipo relativo ao processo anterior. As tecnologias utilizadas foram: QPR, para o desenho do processo; e OutSystems, para a implementação do protótipo.

No capítulo 6, estão apresentados os resultados comparativos entre o processo de Gestão de Alterações do ITIL e o processo de manutenção aplicacional. Esta comparação foi feita com o auxílio de uma ferramenta denominada Savvion, que além de permitir o desenho de processos através da terminologia UML, permite a geração de fluxos sobre as actividades/processos definidos anteriormente.

O capítulo 7 apresenta, por fim, as conclusões deste trabalho e especifica também orientações para um trabalho futuro.

2. Estado da Arte

Este capítulo está subdividido em 3 capítulos: Manutenção de Software, Gestão da Manutenção de Software e *Outsourcing*.

O objectivo de cada um destes capítulos é fazer uma introdução aos conceitos destas 3 áreas, bem como estabelecer o contexto das mesmas no tema desta tese: "A Manutenção de Software num contexto de Full-*Outsourcing*", do ponto de vista da comunidade científica.

2.1 Manutenção de Software

Nos últimos 20 anos tem-se vindo a assistir à proliferação de processos de engenharia de software. Esta proliferação permite às organizações lidar com a inevitável variedade de projectos de software, de um modo flexível, mas enfraquece as suas defesas contra algumas fontes comuns de falha de projectos e não permite pontos à volta dos quais seja possível planear e controlar (Car, et al., 2002).

Alguns dos modelos de ciclos de vida do processo de engenharia de software são: Code-and-Fix, incremental, prototipagem, orientado a objectos, sendo o mais citado pela comunidade o modelo Cascata proposto por Royce em 1970.

Com o decorrer dos anos, os referidos modelos sofreram algumas alterações de modo a se adequarem aos tempos que correm, visto no início da indústria de software não existir diferença entre o desenvolvimento e a manutenção (Bennett, et al., 2000).

Tradicionalmente a manutenção é vista como a última actividade do ciclo de vida do software (Bennett, et al., 2000) pelo modelo em Cascata, proposto por Royce. Mas em certos casos ainda é vista sob a forma das normas como IEEE1074-1997 standard, que representam a manutenção de software como o penúltimo passo dos processos de desenvolvimento de software (April, et al., 2005). Sendo o último o encerramento da aplicação sugerido pelo modelo de Fases (April, et al., 2005).

Vamos abordar ambos os modelos, Cascata e Fases, de modo a enquadrar a fase de manutenção do software no contexto do ciclo de vida do mesmo. Durante a investigação foi muito difícil encontrar metodologias que representassem o processo de manutenção de software, ou as suas actividades.

Bennett denota que os primeiros ciclos de vida de manutenção consistiam em 3 simples actividades: compreensão, modificação, validação da alteração no software; estes estados passam agora a ser as linhas mestras para o processo e não as actividades em si.

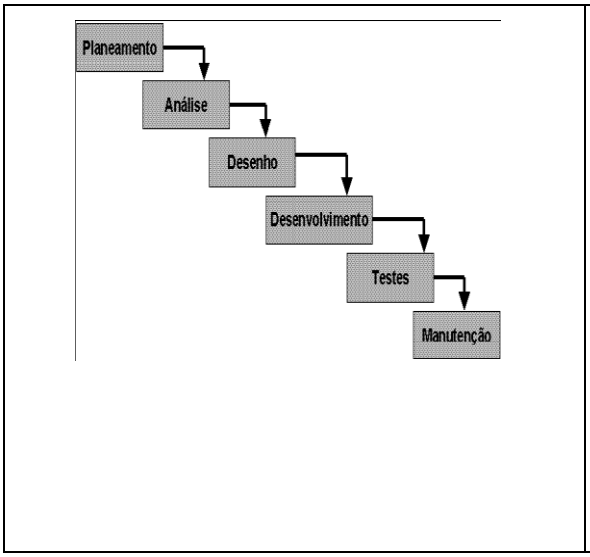
2.1.1. Enquadramento no ciclo de vida do software

Modelo Cascata

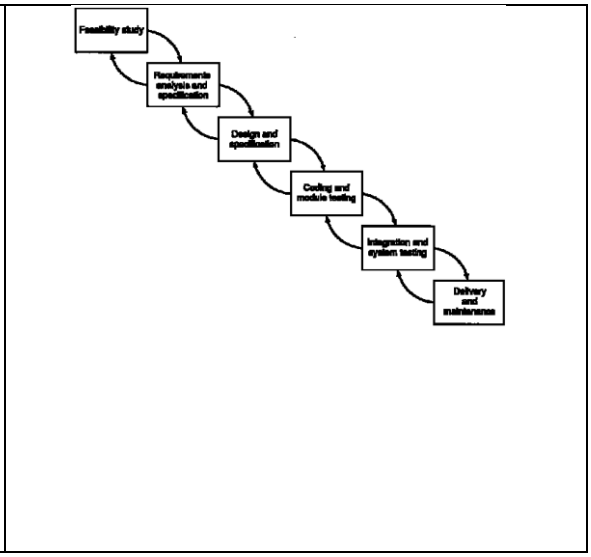
Em 1970, Royce propôs o que é agora popularmente designado como modelo em Cascata, figura 1, como um conceito inicial para a formulação do ciclo de vida do software. O seu trabalho então explorou, como o modelo inicial poderia ser desenvolvido em um modelo iterativo, com feedback de cada fase influenciando as próximas fases, de modo similar a muitos métodos utilizados hoje em dia (Fig. 2). Ironicamente, foi somente o modelo inicial que mereceu destaque. O modelo em Cascata não se tornou o que Royce pretendia: num projecto iterativo, mas sim num modelo puramente sequencial e ordenado.

No modelo em Cascata a evolução do software respeita uma sequência ordenada de níveis, cujo objectivo era ajudar empresas na gestão de projectos de desenvolvimento de software, bem como na gestão da equipa associada ao projecto.

Em ambos os modelos podemos ver que as actividades são executadas sequencialmente, de forma que uma tarefa só tem início quando a tarefa anterior tiver terminado.



Fig, 1- O Modelo em Cascata Simples (in Wikipedia)



Fig, 2- Modelo em Cascata iterativo (*in* (Boehm, 1988))

As actividades compreendidas no Modelo de Cascata tradicional são as seguintes:

Análise e definição dos requisitos: nesta etapa, estabelecem-se os requisitos do produto que se deseja desenvolver, o que consiste nos serviços que se devem fornecer, limitações e objectivos do software. Sendo isso estabelecido, os requisitos devem ser definidos de uma maneira apropriada para que sejam úteis na etapa seguinte.

Esta etapa inclui também a documentação e o estudo da facilidade e da viabilidade do projecto com o fim de determinar o processo de início de desenvolvimento do projecto do sistema; pode ser vista como uma concepção de um produto de software e também como o início do seu ciclo de vida.

Desenho do sistema: o desenho do sistema é um processo de vários passos que se centraliza em quatro atributos diferentes do sistema: estrutura de dados, arquitectura do software, detalhes procedimentais e caracterização das interfaces. O processo de desenho representa os requisitos de uma forma que permita a codificação do produto. Da mesma maneira que a análise dos requisitos, o desenho é documentado e transforma-se em uma parte do software.

Desenvolvimento: esta é a etapa em que são criados os programas. Se o projecto possui um nível de detalhe elevado, a etapa de codificação pode implementar-se automaticamente. A princípio, sugere-se incluir um teste unitário dos módulos nesta etapa; nesse caso, as unidades de código produzidas são testadas individualmente antes de passar a etapa de integração e teste global.

Testes do sistema: concluída a codificação, começa a fase de teste do sistema. O processo de teste centraliza-se em dois pontos principais: as lógicas internas do software e as funcionalidades externas. Esta fase decide se foram solucionados erros de comportamento do software e assegura que as entradas definidas produzam resultados reais que coincidam com os requisitos especificados.

Manutenção: esta etapa consiste na correcção de erros que não foram previamente detectados e envolve mudanças no software com a finalidade de corrigir defeitos e deficiências encontradas durante o uso do software bem como a adição de novas funcionalidades de modo a melhorar a usabilidade e a aplicabilidade do software.

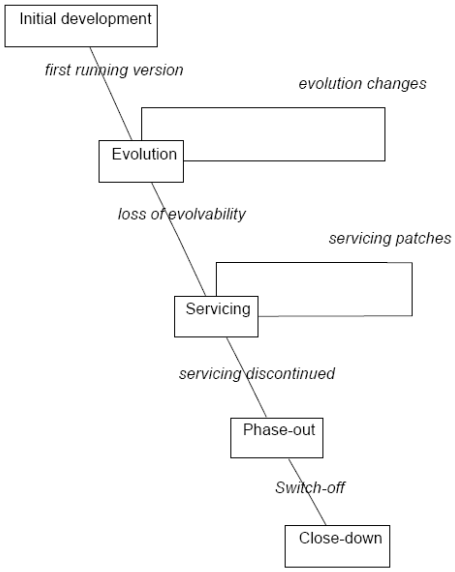
Todas as variações do modelo Cascata possuem o mesmo conceito básico: a ideia de que uma etapa fornece a saída que será usada como entrada para a etapa seguinte. Portanto, o processo de desenvolvimento de um produto de software de acordo com o modelo Cascata é simples de conhecer e controlar.

Outras actividades também levadas em consideração em cada uma das etapas do ciclo de vida do software, são as seguintes: a documentação, a verificação e a administração das etapas. A verificação, por sua vez, é necessária para que uma etapa forneça os dados correctos para a etapa seguinte. Já a administração, efectua a gestão e o controle da etapa. A actividade de documentação origina um conjunto de documentos resultantes de cada etapa do ciclo.

Modelo de Fases

Em contraste, com as definições standard oferecidas na secção anterior, onde a manutenção é vista como uma única actividade pós-entrega, um novo modelo é introduzido por Bennett (Fig. 3). Este modelo contém cinco estados diferentes: desenvolvimento inicial, evolução, serviço, encerramento e fecho (Bennett, et al., 2000), 47].

O modelo em fases foi introduzido em 1999 e está resumido na Figura 3. Este modelo representa o ciclo de vida do software como uma sequência de fases, sendo a primeira o desenvolvimento inicial. O seu contributo chave é a separação do nível de manutenção numa fase de evolução, seguida de uma fase de serviço e encerramento.



Fig, 3- Modelo de Fases Simples (*in* (Bennett, et al., 2000))

As fases compreendidas pelo Modelo de Fases Simples são as seguintes:

Desenvolvimento inicial: nesta fase é desenvolvida a primeira versão funcional do sistema. Tem como objectivo encontrar meios de desenvolvimento de modo a que o software possa ser alterado de forma fácil e segura nas fases subsequentes. Os parâmetros de saída desta fase são: a arquitectura e o conhecimento da equipa.

Evolução: nesta fase são aumentadas as capacidades e funcionalidades do sistema para ir de encontro com as necessidades dos utilizadores, ao nível mais profundo possível. A evolução apenas acontece quando o desenvolvimento inicial é realizado com sucesso. O objectivo é adaptar a aplicação para qualquer alteração de requisitos do utilizador e do ambiente operacional.

A fase de evolução corrige também as falhas na aplicação e reporta-as ao colaborador e ao utilizador, podendo estes fazer exigências mais exactas baseadas na experiência passada com a aplicação.

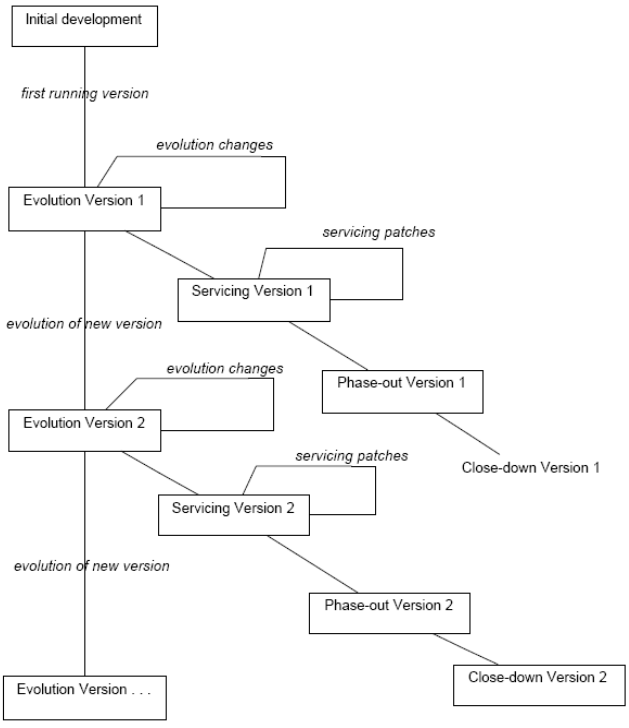
Serviço: são feitas reparações a pequenos defeitos e alterações funcionais simples. O objectivo é executar e testar mudanças técnicas ao software, empreendendo isto no custo mínimo e dentro das potencialidades da equipa de funcionários disponível. A passagem para a fase de serviço envolve uma grande mudança na gestão de projecto: somente as correcções menores, os realces e o trabalho preventivo devem ser empreendidos; os desenhadores e os arquitectos seniores não necessitam estar disponíveis; a equipa de funcionários não requer o mesmo nível de domínio da engenharia ou da perícia da tecnologia de programação.

Encerramento: a empresa decide não continuar a providenciar o serviço, procurando tirar proveito do sistema o maior tempo possível.

Fecho: a empresa retira o sistema do mercado e direcciona os utilizadores para um sisrema substituto, caso este exista.

Modelo de Fases com versões

A existência deste modelo demonstra como podemos fazer as fases do modelo durar mais tempo, dando deste modo mais um nível de estudo e aumentando a esperança média de vida de um programa, como se pode ver na figura 4.



Fig, 4- Modelo de Fases com versões (*in* (Bennett, et al., 2000))

Alterações no software

A alteração de software é a operação básica quer da fase de evolução, quer da fase de serviço. As duas fases estão separadas pela dificuldade inerente da alteração.

Alterações substanciais são permitidas na fase de evolução, enquanto que na fase de serviço as alterações permitidas são limitadas; no entanto, tanto a fase de evolução como de serviço é derivada da necessidade de alterações sucessivas. Os grandes assuntos de evolução de arquitecturas e deterioração de código, ganham com o estudo detalhado das propriedades da alteração individual. Os processos e métodos agora descritos irão beneficiar de pesquisas futuras que levarão a um benefício industrial substancial.

Uma alteração é um processo que introduz novos requisitos ao sistema existente, ou modifica o sistema se os requisitos não tiverem sido implementados correctamente, ou move o sistema para um novo ambiente operacional.

O mini ciclo da alteração consiste nas seguintes fases: requisito de alteração, fase de planeamento, compreensão do programa, análise de impacto da alteração, implementação da alteração, reestruturar para a alteração, propagação da alteração, verificação e validação e re-documentação.

Um pedido de alteração é muitas vezes originado pelos utilizadores do sistema, e pode ter a forma de um reporte de erros ou de um pedido de funcionalidade adicional.

2.1.2. Definições da manutenção de software

A manutenção compreende um leque de conceitos que são essenciais para o entendimento da sua representação e para clarificar as definições utilizadas neste documento que serão as seguintes (Boehm, 1988) (Mamone, 1994):

- *Manutenção*: modificação de um produto de software após a entrega para corrigir falhas, melhorar o desempenho ou outros atributos, ou para adaptar o produto a um ambiente da mudança;
- *Evolução*: a evolução de software prende-se com a saída de uma nova versão, ou seja uma versão do produto com um conjunto relevante de alterações;
- *Alteração*: mudança do código software. Engloba a correcção de erros e a satisfação de novos requisitos propostos pelo utilizador;
- *Melhoramento*: mudança no software de modo a melhorar a performance do produto, ou a adaptá-lo a novos ambientes;

- *Processo de manutenção*: define as actividades do programador, da organização que providencia os serviços de manutenção do software, ou seja os serviços de manutenção para deixar o software actualizado. O objectivo deste processo é modificar um produto de software existente, preservando a sua integridade;
- *Manutenção de software*: modificação de um produto de software após a entrega para corrigir falhas, para melhorar o desempenho ou outros atributos ou para adaptar o produto a um ambiente modificado.

2.1.3. Categorias da manutenção de software

Ao longo dos anos, o estudo sobre MS tem vindo a ser cada vez mais completo e as pessoas têm cada vez mais interesse no assunto, por isso a categorização foi sendo aperfeiçoada especialmente nos últimos 20 anos.

Lientz e Swanson

No final da década de 70, Swanson e Lientz apresentaram 3 categorias de manutenção (Swanson, 1976), (Lientz, et al., 1980):

- *Adaptativa*: Resposta a alterações em ambientes de dados ou de processamento;
- *Correctiva*: Resposta a falhas de vários tipos;
- *Evolutiva*: Executada para eliminar ineficiências de processamento, aumentar o desempenho, ou melhorar a capacidade de manutenção.

Bennett

Enquanto que a adaptativa e a correctiva têm como função manter o programa operacional, a evolutiva tem de manter o programa operacional ao menor custo ou de modo a servir melhor as necessidades dos seus utilizadores (Bennett, et al., 2000).

Num trabalho recentemente publicado em 2000, Bennett faz referência a Lientz e Swanson dizendo que estes últimos categorizam as actividades de manutenção em 4 tipos (Bennett, et al., 2000). A quarta actividade advém da separação da categoria Evolutiva em 2 actividades independentes. Deste modo as 4 actividades são:

- *Adaptativa*: mudanças no ambiente de software;
- *Correctiva*: reparação de erros;
- *Evolutiva*: novos requisitos do utilizador;
- *Preventiva*: prevenir problemas futuros.

Norma ISO/IEC

Como era de esperar as normas foram actualizadas e foi admitida mais uma categoria pela ISO/IEC (IEEE, 1998). Deste modo as categorias de manutenção ficaram definidas como:

- *Adaptativa*: modificação de um produto de software após a sua entrega de modo a mantê-lo operacional num ambiente mudado ou em mudança;
- *Correctiva*: modificação de um produto de software após a sua entrega para corrigir problemas descobertos;
- *Evolutiva*: modificação de um produto de software após a sua entrega para melhorar a sua performance ou sua capacidade de manutenção;
- *Preventiva*: modificação de um produto de software após a sua entrega para detectar e corrigir faltas latentes, antes que se tornem faltas efectivas.

2.1.4. Motivação

A manutenção consome a maior parte do custo do ciclo de vida do software. Os gastos em manutenção são mencionados na maioria dos surveys. Os valores oscilam entre os 40% e os 80%, como mostra a Tabela 1.

Autores	Esforço em manutenção
Lientz , Swanson	>40%
Lientz , Swanson	½ Homem/Ano
Boehm	60%
Pigoski	60% a 80%
Swanson, Beath	50% a 70%

Tabela 1 -. Esforço dispendido na manutenção de software

Os valores acima apresentados são divididos pelas categorias da manutenção da seguinte forma:

- A manutenção evolutiva e adaptativa detêm 75% do esforço e a manutenção correctiva detém cerca de 21% (Bennett, et al., 2000);
- A área que de longe ocupa o maior esforço a nível de manutenção é a evolutiva, utilizando 2/3 do esforço dispendido nas 4 áreas da manutenção (Boehm, 1988).

Os estudos mostram que a incorporação de novos requisitos (pedidos por parte dos utilizadores) são o núcleo do problema da evolução e manutenção de software, visto que o tempo médio de vida de uma aplicação na fase de manutenção é de 8 a 10 anos o esforço gasto nesta fase tem de ser elevado só por si (Bennett, et al., 2000).

A manutenção é necessária para assegurar que o sistema continua a satisfazer as necessidades dos utilizadores (IEEE, 1998). Uma visão comum da manutenção é que esta serve para arranjar pequenos defeitos. Contudo, estudos ao longo dos anos mostram que cerca de 80% do esforço de manutenção é aplicado em acções *não-correctivas* (IEEE, 1998).

Esta percepção é concebida a partir do momento em que os utilizadores submetem relatórios com problemas, que na verdade são aumentos de funcionalidade ao sistema, ou seja propostas de evolução. A inclusão destes melhoramentos nos relatórios de problemas leva à confusão de ambos os conceitos no que diz respeito à manutenção.

Durante o tempo de vida do software, este é submetido a diversas alterações, as quais podem ser causadas por uma enorme variedade de razões e necessidades.

Normalmente, é impossível definir toda a aplicação desde o início, mesmo assim as actividades de desenvolvimento começam. Diversos detalhes do processo podem, e por vezes têm de, ficar por especificar inicialmente, e são expressos de forma mais detalhada quando informação adicional, ou mais precisa, da aplicação tenha sido recolhida e analisada.

Por exemplo, os procedimentos de teste só podem ser desenhados quando os requisitos de qualidade da aplicação a ser desenvolvida sejam especificados. Este tipo de alteração é denominada definição incremental, visto adicionar novas partes ao processo. Estas novas alterações não devem interferir com o modelo existente, senão, seria necessário alterar a parte restante.

Os processos de software são entidades que estão em ambientes altamente dinâmicos. Uma aplicação pode ter de mudar como resposta a alterações no ambiente no qual opera ou alterações na organização ao nível corporativo. Tais alterações podem ser causadas por desempenho fraco, novas ferramentas adquiridas pela organização para auxiliar os programadores de desenvolvimento de software, alterações na estratégia de marketing, ou alterações do cliente quer a nível de expectativas quer a nível de requisitos. Assim, um processo existente tem de ser modificado ou estendido para reflectir a evolução de ambiente e/ou alterações internas.

Tem de ser possível costumizar um processo de modo a permitir um agente seleccionar a solução mais eficaz para um determinado problema. Por exemplo uma actividade de desenho pode ser conduzida segundo uma de duas abordagens, ascendente ou descendente, dependendo das características do sistema que irá ser desenhado e do juízo do agente.

Existem diversas políticas que podem ser adoptadas para a aplicação de alterações numa determinada aplicação. Num extremo temos as alterações que são aplicadas a especificações do processo, que não afectam a definição das actividades. Os efeitos são modificações estáticas que se tornam visíveis apenas quando é lançado um processo que seguiu o modelo modificado.

No outro extremo, temos uma política altamente dinâmica e alterações as especificações do processo são imediatamente propagadas às actividades afectadas.

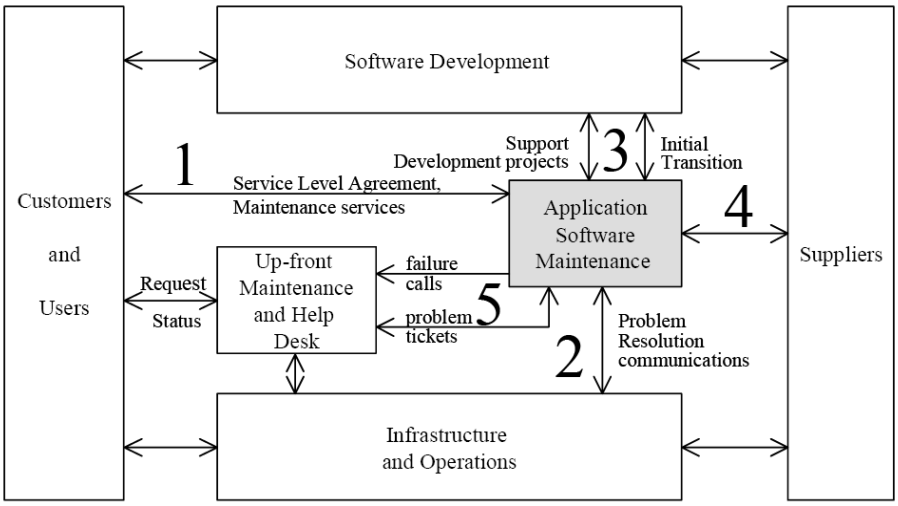
Características chave na natureza e tratamento de pequenos pedidos de manutenção, tem vindo a sobressair (April, et al., 2005), devido aos seguintes factos:

- Os pedidos da modificação são reportados mais ou menos aleatoriamente e não podem ser esclarecidos individualmente no processo de plano anual de orçamento;
- Os pedidos de modificação são revistos e as prioridades atribuídas; frequentemente no nível operacional a maioria destes pedidos não requerem a participação da gestão sénior;
- O trabalho de manutenção não é controlado usando técnicas de gestão de projecto, mas sim através de técnicas de gestão de filas;
- O tamanho e a complexidade de cada pedido da manutenção são de tal forma pequenos que podem geralmente ser assegurados por um ou dois recursos;
- As prioridades podem ser alteradas em qualquer altura, e um pedido de modificação ou de correcção de aplicação pode passar a ter prioridade sobre o outro trabalho em andamento.

2.2. Gestão da Manutenção de Software

É referido por diversos autores que muitas organizações de software não têm processos definidos para as suas actividades de manutenção. Van Bon confirma a falta de processos de gestão na área de manutenção de software e que esta é muito negligenciada (April, et al., 2005)

É importante perceber a janela das actividades de manutenção e o contexto no qual o programador de software opera no seu dia-a-dia. A figura 5 resume as interfaces existentes na manutenção de software num contexto organizacional típico.



Fig, 5- Diagrama de Contexto da Manutenção Software (*in* (April, et al., 2005))

As interfaces existentes com a Manutenção de Software são as seguintes:

- *Interface 1*: clientes e utilizadores de manutenção de software;
- *Interface 2*: departamento de infra-estruturas e operações;
- *Interface 3*: programadores;
- *Interface 4*: fornecedores;
- *Interface 5*: manutenção de primeira linha e help-desk.

Tendo em conta estas interfaces que requerem serviços diários, a gestão da manutenção tem de manter as aplicações: a correrem suavemente, a reagir rapidamente de modo a restaurar o serviço, a corresponder ou exceder o nível de serviço acordado e ainda manter a comunidade de utilizadores confiante que existe uma equipa competente e dedicada à sua disposição que actua dentro do orçamento acordado.

Cada interface designada tem um objectivo:

A primeira interface lida com os clientes e utilizadores e é muito importante, visto que consiste na negociação e discussão acerca de: pedidos individuais de prioritização, acordos de níveis de serviço (SLAs), planear, orçamentação, serviço ao cliente e actividades relacionadas com a satisfação dos utilizadores. Os utilizadores operam no software e serão frequentemente envolvidos em comunicações diárias que requerem:

- Resposta operacional rápida a relatórios de problemas;
- Habilidade de resposta a questões referentes a regras de negócio, ecrãs ou relatórios específicos do negócio;
- Relatórios de progresso num grande numero de pedidos de modificação.

A segunda interface lida com a comunicação da empresa quer a nível operacional quer a nível de infra-estruturas (ITSMF-NL, 2005). Estas são responsáveis pela infraestrutura de suporte às aplicações. Elas lidam com todo o suporte e manutenção associadas com postos de trabalho, com redes e plataformas, bem como conduzem actividades como a criação de cópias de segurança, recobro, e administração de sistema. O utilizador raramente está a par, ou envolvido, na troca interna de informação entre os programadores de software e as operações. Esta interface também inclui actividades menos frequentes como coordenação e recuperação de serviço após falhas ou desastres, com vista a restaurar o acesso a serviços acordados nos termos e condições dos SLA.

A terceira interface está localizada entre o desenvolvimento e a manutenção, e é tipicamente iniciada durante o desenvolvimento de novo software. A causa inicial de vários problemas de manutenção pode ser monitorizada até ao desenvolvimento, e está reconhecido que os programadores necessitam de estar envolvidos e exercer alguma forma de controlo durante a pré-entrega e a transição (April, et al., 2005).

Também nesta interface de desenvolvimento-manutenção as contribuições feitas pelos programadores que de modo concorrente dão suporte, e por vezes estão envolvidos, também em grande número de projectos em desenvolvimento. O conhecimento do programador de software e portfolio de dados é de grande valor para os programadores de manutenção que precisam de interface com software obsoleto.

A quarta interface mostra a relação com um número crescente de fornecedores, fornecedores de *outsourcing*, e vendedores de *Enterprise Resource Planning* (ERP) (April, et al., 2005), (April, et al., 2001). Os programadores têm diferentes tipos de relações com os fornecedores.

Para assegurar um serviço de qualidade aos utilizadores, os programadores têm de desenvolver alguma compreensão dos variados tipos de contrato e geri-los de forma eficiente de modo a assegurar a performance do fornecedor, que normalmente tem impacto nos resultados dos SLA (Service Level Agreements).

A última interface pode ser representada de diversas formas, de acordo com as diferentes estruturas organizacionais. O *help-desk* é fundado, por vezes, como parte da organização de manutenção, ou parte das operações da organização e pode ser localizado num produto independente do suporte à organização.

Foi observado que alguns utilizadores passam a primeira linha de suporte e acedem directamente ao pessoal da manutenção. Para ser eficaz, é usado um processo mecânico de resolução de problemas que assegura a comunicação para uma rápida resolução de falhas. Um pedido específico de um utilizador, pode ser chamado de *ticket*, e tipicamente circula entre o *help-desk*, a manutenção e o nível operacional que pode a isolar o problema (April, et al., 2001).

2.2.1. Implementação

Nos próximos sub-capítulos estão descritas duas filosofias sobre a Manutenção de Software, numa perspectiva de serviço. A primeira é o ITIL (*Information Technology Infrastructure Library),* uma estrutura com um conjunto de directrizes que visa ajustar pessoas, processos e tecnologia. No ITIL esta descrita uma configuração para a realização da manutenção de software de uma forma orientada ao serviço. A segunda é o Modelo de Maturidade de Capacidade de Serviço de TI, que é um modelo a partir do qual de pode avaliar o nível de maturidade dos processos de TI, nomeadamente o processo de manutenção, numa empresa. Este modelo é análogo ao Modelo de Maturidade de Capacidade de Software, originado pelo CMM. Ao nível individual dos processos, ambos os modelos (MMCSTI e CMMS) têm muito em comum.

2.2.2. ITIL

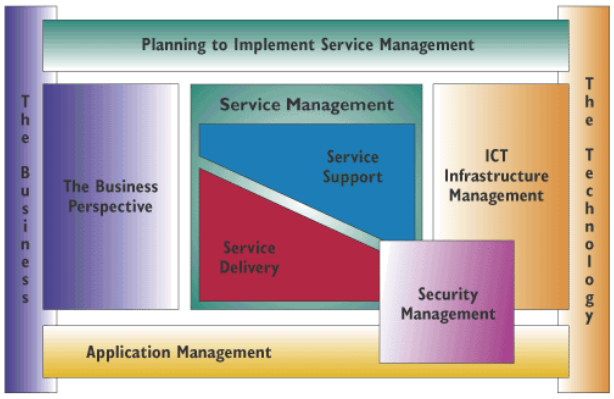
A definição do ITIL (ITSMF-NL, 2005) (Office of Government Commerce, 2002) quer dizer Biblioteca de Infra-estruturas de Tecnologias de Informação. Esta secção vai tentar mostrar a parte desta biblioteca que se dedica a gestão das alterações, bem como mostrar o seu enquadramento.

O ITIL é considerado como as melhores práticas de tecnologia de informação a nível europeu, cujo objectivo principal é estabelecer melhores práticas e um standard de qualidade de serviços TI, que os clientes têm de exigir e que os prestadores de serviço devem tentar fornecer CCTA (1993).

O ITIL foi desenvolvido originalmente pelo governo britânico através da CCTA (*Central Computer & Telecommunications Agency*). Hoje em dia, o ITIL é mantido pelo EXIN (*Netherlands IT Examinations Institute*).

Enquadramento

O ITIL materializa-se num conjunto de manuais que documentam as actividades no fornecimento de serviços de TI, como se pode ver na figura 6. A área de que mais relevância apresenta para este estudo é a Gestão de Serviços, que enquadra duas grandes áreas: Suporte aos Serviços e Disponibilização de Serviços.



Fig, 6- A estrutura de publicações do ITIL (*in* (ITSMF-NL, 2005))

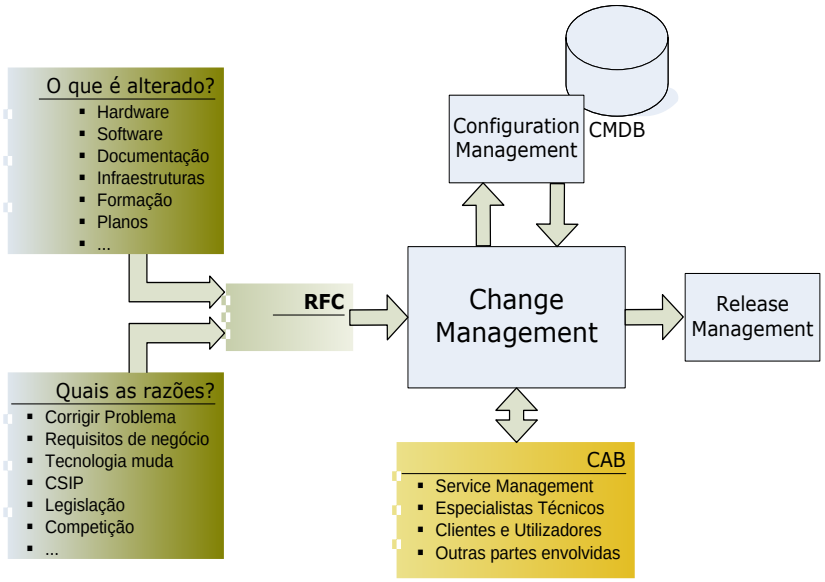
Tanto o Suporte como a Disponibilização dos Serviços são compostos por um conjunto de actividades, e ambos lidam com o tema da manutenção, mas em perspectivas diferentes. No lado do Suporte temos como actividade principal a Gestão de Alterações e no lado da Disponibilização temos a Gestão de Níveis de Serviço.

Gestão de Alterações

A gestão de alterações tem como objectivo assegurar que os métodos e procedimentos normalizados são utilizados para o tratamento rápido e eficaz de todas as alterações para minimizar o impacto de eventuais incidentes.

As alterações nas infra-estruturas das TI podem surgir, de modo reactivo, em resposta a problemas ou requisitos impostos externamente, como, por exemplo, alterações legislativas; ou então em modo pró-activo, da busca de maior eficiência e eficácia ou para permitir, ou reflectir, iniciativas de negócio.

A Gestão de Alterações é responsável pelo controlo das alterações de todos os CIs (elementos de configuração que residem na CMDB) no ambiente de produção (AP). Não é responsável pelas alterações em projectos em curso, uma vez que, estas são controladas pelo processo de alterações de projectos. No entanto, espera-se estreita ligação entre os gestores do projecto de desenvolvimento e o Gestor de Alterações.



Fig, 7- Diagrama de Contexto da Gestão de Alterações (*in* (ITSMF-NL, 2005))

Como mostra a figura 7, a informação de entrada da Gestão das Alterações é um Pedido de Alteração (RFC). Os RFC podem ter 3 origens:

- *Standard*: alteração comum, com condições para autorização e aprovação de custo preestabelecidas;
- *Normal*: Alteração não standard e não urgente, que precisa ser processada e avaliada pela Gestão de Alterações.
- *Urgente*: Alteração demasiado urgente para seguir os processos normais da Gestão de Alterações.

Existem dois processos para realizar uma alteração: normal e urgente. O procedimento de Alteração urgente não deve ser encarado como um percurso opcional, para uma implementação mais rápida, uma vez que comporta riscos consideráveis em relação ao procedimento de alteração normal.

Uma alteração normal é uma solução aceite para um conjunto de requisitos identificável e relativamente comum, em que é efectivamente dada autoridade antes da implementação, como por exemplo, a configuração de perfis de acesso para um novo colaborador. A Gestão de Alterações deve ser integrada na gestão de grandes projectos ou programas empresariais, mediante planeamento, construção, teste e implementação.

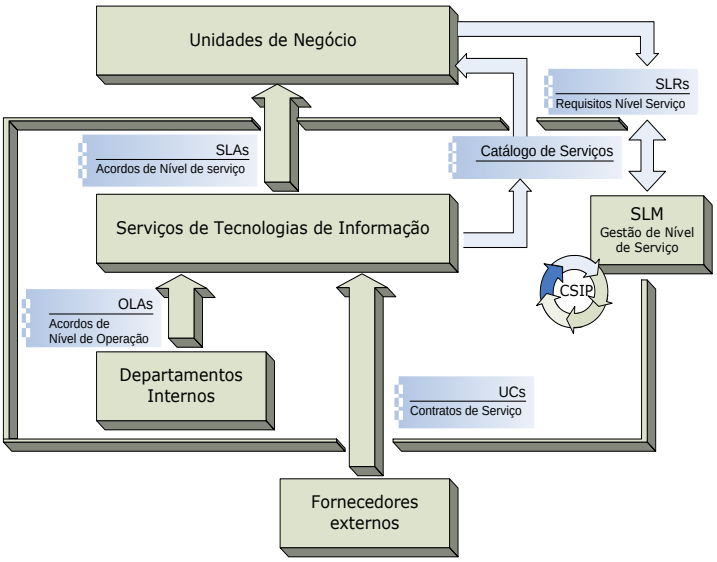
A actividade de Gestão de Alterações é claramente base de aplicação dos conceitos mencionados no capítulo 2.

Gestão de Níveis de Serviço

O objectivo da Gestão de Níveis de Serviço é tornar claro acordos com o cliente, destinatário do serviço, acerca do tipo e da qualidade dos serviços de TI a serem fornecidos e implementar esses mesmos acordos.

Garantir a melhoria progressiva da qualidade dos serviços de TI, bem como o seu alinhamento com o negócio, através de um processo contínuo de negociação, monitorização, elaboração de relatórios e revisão dos serviços de TI fornecidos, instigando acções para eliminar níveis de serviço inaceitáveis.

A Gestão de Níveis de Serviço (SLM) garante que os objectivos dos serviço são acordados e documentados em Acordos de Nível de Serviço (SLAs), além de controlar e rever os níveis de serviço existentes face aos objectivos dos SLAs. A SLM deverá igualmente tentar melhorar, proactivamente, todos os níveis de serviço no âmbito das restrições de custos impostas.



Fig, 8 - Gestão de Níveis de Serviço (*in* (ITSMF-NL, 2005))

A figura 8 é explicativa de todo o processo, e claramente o apoio para todo o processo já descrito em 3.1.

Este processo é o de mais relevância na Disponibilização de Serviços, visto ser base para o que na comunidade de investigação se fala do processo de manutenção com um fornecedor que pode ser um departamento interno de serviços, ou uma empresa externa subcontratada ou ainda um fornecedor de terceiros, visto no capítulo 3.1.

2.2.3. Modelo de Maturidade de Manutenção de Software numa perspectiva de serviço

O Modelo de Maturidade de Software de Serviços de TI (SMmm) de Niessink, e van Vliet, é um modelo de crescimento de maturidade análogo ao Modelo de Maturidade de Software (CMM). Este modelo retém a referência aos níveis de maturidade do processo pai (April, et al., 2005).

A estrutura do modelo é similar ao CMM, mas o seu domínio de aplicação é diferente. Enquanto que o CMM engloba o processo de desenvolvimento de software, o SMmmI abrange os processos chave para produzir serviços de TI de alta qualidade. É importante notar que o CMM diz-se adequado para ambos, desenvolvimento e manutenção, no entanto foram encontradas dificuldades em implementar este modelo numa organização que apenas lide com a manutenção.

Este modelo mede a maturidade das organizações numa escala de 5 níveis, muito semelhante à escala proposta pelo CMM. Os níveis de maturidade correspondem à capacidade da empresa em realizar projectos grandes e complexos e estão classificados da seguinte forma (April, et al., 2005):

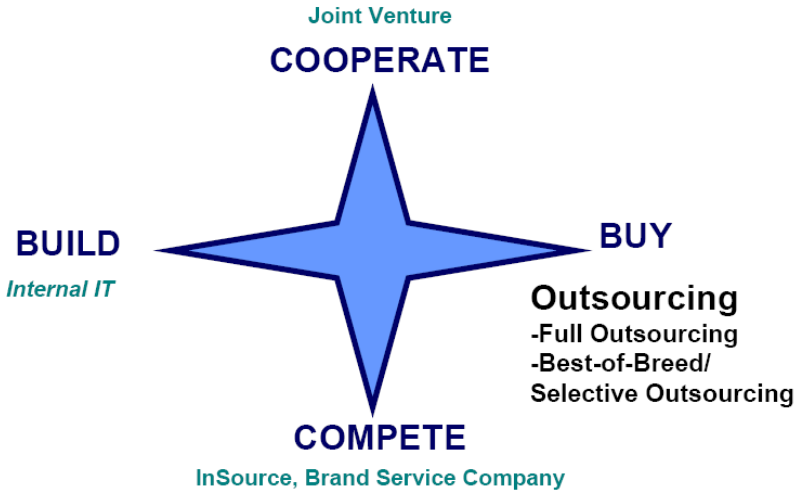
- Nível 1. Inicial – a empresa possui processos ad-hoc e caóticos. Poucos processos estão definidos, e o sucesso depende do esforço individual;
- Nível 2. Repetitivos – Os processos da gestão do serviço básico são estabelecidos para seguir o custo, a programação e o desempenho das TIs na entrega do serviço. Estes processos são caracterizados por projectos, e muitas vezes ainda trabalham de forma reactiva;
- Nível 3. Definido – a empresa possui processos standard definidos, documentados para a organização. Normalmente a empresa trabalha de forma activa;
- Nível 4. Gestão – a empresa mede e controla os seus processos, devido à recolha de medidas detalhadas da prestação de serviços de manutenção ao processo da entrega e a qualidade do serviço.
- Nível 5. Optimização – a empresa tem em foco descobrir a causa dos seus problemas e melhorar continuamente os seus processos.

O SMmm avalia a maturidade de unidades organizacionais específicas de uma empresa e se esta estiver num nível de maturidade, significa que implementa todas as áreas de processo do respectivo nível.

2.3. *Outsourcing* dos SI

Este capítulo aparece neste estudo devido a várias referências durante todo o trabalho. São vários os *surveys* que referenciam a possibilidade de recorrer ao *outsourcing* para realizar a manutenção, mas nenhum opta por aprofundar esse tema. Ou seja, a forma como o *outsourcing* encaixa no capítulo da manutenção.

Na realidade, existem várias opções de parceria entre as empresas de Sistemas de Informação, das quais vamos dar mais ênfase ao *outsourcing*.



Fig, 9- Opções de Sourcing (*in* (Gartner, 2002))

Várias empresas de vanguarda optaram por esquemas de *outsourcing* total, preocupando-se apenas com a gestão dos seus sistemas, como por exemplo a EDP. No entanto existem vários esquemas de parcerias disponíveis consoante a estratégia das organizações.

2.3.1. Outsourcing

O *Outsourcing* é a prática de usar empresas exteriores para assegurar trabalho que se executa normalmente dentro de uma empresa. Existem empresas que fazem *outsourcing* de contabilidade, distribuição, manutenção e muitas outras funções importantes – frequentemente porque não têm nenhuma outra escolha. Muitas empresas escolheram colocar determinados processos em *outsourcing*, e como resposta indústrias inteiras evoluíram no sentido de servir essa mesma necessidade.

Como se pode ver na figura 9, a existência de *outsourcing* implica a compra/venda de um produto, o qual pode ser um serviço, uma tecnologia, entre outros. Ou seja, *outsourcing* designa a acção que existe por parte de uma organização em obter mão-de-obra de fora da empresa, de modo a realizar a subcontratação de serviços.

Em 2004, o mercado de *outsourcing* de TI cresceu 37%, mas foi caracterizado por um elevado número de pequenos negócios e uma grande concorrência por parte dos vendedores (Pankaj, et al., 2006).

O crescimento deste mercado deve-se ao facto das empresas estarem finalmente a aperceber-se das seguintes realidades:

- Apenas 25 a 33% do esforço total gasto durante o ciclo de vida do software vai realmente para a construção do sistema e que o resto é consumido pelo esforço gasto com manutenção operacional do sistema (Pankaj, et al., 2006);
- Os sistemas de software estão a tornar-se cada vez mais complexos, quer a nível funcional quer a nível técnico. Deste modo essa complexidade acrescida complica a estimação do esforço, a qualidade e maturidade do software sob a responsabilidade da manutenção, também tem um grande peso na estimação dos esforços de manutenção.

Tipos de *Outsourcing* (Gartner, 2002)

O *outsourcing* divide-se em 2 actividades:

- *Outsourcing* total - que concentra a operação num único fornecedor, por oferecer ganho de escala e propostas mais económicas.
- *Outsourcing* selectivo ou múltiplo – o melhor fornecedor para cada uma das actividades realizadas em *outsourcing*.

***Outsourcing* de TI**

O processo de *outsourcing* de TI envolve gestão e operacionalidade eficazes de recursos da tecnologia de informação de corporações de uma posição remota. Para explicar este conceito, podemos considerar o exemplo da gestão de uma rede de computadores de uma grande corporação que inclua a monitorização de falhas, a monitorização de alarmes, a monitorização remota de centros de dados, entre outros.

A natureza destas operações é complexa e requer um suporte em tempo real sempre que precisa. Posto isto, este tipo de processo não pode ser facilmente efectuado em regime de *outsourcing* em locais onde há uma grande diferença horária. Tais tarefas são atribuídas às empresas com o mesmo fuso horário.

2.3.2. Programadores de Manutenção

A actividade de manutenção é frequentemente desempenhada por equipas que não estiveram envolvidas no desenho e desenvolvimento do software. Isto é evidente, visto que é facto que uma parte significante do trabalho de *outsourcing* de TI é relativo a manutenção de sistemas de software existentes. Os programadores de manutenção recebem dos clientes o software e a sua documentação, quando existente. Sempre que um programador é substituído, quer seja por doença ou despedimento, o seu substituto terá de perder tempo a compreender o código. Estes casos não estão previstos no contrato entre o cliente e o outsourcer, tendo este último de colmatar essas falhas de modo a cumprir todos os níveis de serviço acordados.

2.3.3.Técnicas de Manutenção

É facto que os programadores gastam tempo a ler e a compreender os programas de modo a poderem implementar alterações nos mesmos.

Browsers de código são ferramentas chave na compreensão de programas, porque ajudam à navegação dentro do código e ainda permitem relacionar as suas partes de uma forma muito mais rápida do que manualmente.

A re-documentação do software é também um passo importante. O objectivo desta actividade é actualizar a documentação, por exemplo quando são efectuadas alterações no software, e torná-la mais clara e explicativa, de modo a diminuir o tempo gasto na compreensão dos programas.

2.3.4. Riscos

Os riscos de recorrer ao *Outsourcing*, por parte dos clientes são, sumariamente, os seguintes:

- Resultado nem sempre correspondente ao esperado, interferindo na qualidade planeada para os clientes;
- Custos por vezes ficam além do previsto; Existência de custos e complexidade na gestão da relação com o outsourcer.
- Dependência de parceiros, ou existência de parceiros pouco fiáveis.

2.3.5. Benefícios

Os benefícios para o cliente quando recorre ao *outsourcing* são, os seguintes:

- Controlo e visibilidade dos custos, pode não ser a única razão mas é certamente um factor principal. O *outsourcing* converte custos fixos em custos variáveis e permite evitar grandes despesas nos estados mais avançados de um negócio.
- Acessos a novos recursos humanos e tecnologia. Permite obter equipas dedicadas em pouco espaço de tempo e de curta duração, assim como pode alojar os recursos onde mais convier para o negócio.
- Transparência no estabelecimento de prioridades e Objectividade na negociação;
- Transferência do risco de parte da actividade para terceiros.

2.3.6. Estimação do Custo

O *outsourcing* de manutenção de software é uma actividade que tem estado a crescer, e com ela cresce também o nível de competitividade. Antes, os clientes procuravam companhias de serviços de TI numa base de tempo-material necessários (onde os serviços eram pagos pelo número de engenheiros envolvidos nas actividades de manutenção). Agora vemos uma mudança definitiva nesta tendência, vendo os clientes exigir SLAs baseados em preços fixos para a manutenção de software durante um certo período de tempo. Muitas vezes o cobrador pelo serviço não possui conhecimentos do estado em que se encontra o software do lado do cliente ou mesmo qual o tipo de software. Posto isto, é importante ter uma estimativa base competitiva baseada em factores que tenham certo peso na actividade de manutenção de software.

3. Problema

Do grande tema atrás exposto (Manutenção de Software) o meu estudo centrar-se-á acima de tudo no desenho do processo. A razão desta escolha prende-se com o facto de, após investigação, o ter considerado ineficaz e ineficiente por isso necessitado de resolução.

De seguida é descrito o problema e os factores que o potenciam, ambos identificados ao longo dos capítulos de apresentação do tema do estudo.

3.1. Factores que potenciam o problema

3.1.1. Rotatividade de pessoas

O estudo realizado nos primeiros capítulos, permitiu a identificação de um factor/motivo base que potencia a maioria dos problemas descritos no que diz respeito à Manutenção de Software: esse factor é a rotatividade de pessoas.

Na maioria dos projectos de software, cada fase do ciclo de vida é desempenhada por equipas diferentes. Logo a passagem de informação entre as várias fases tem de ser perfeita, para quando chegar a fase da manutenção a informação passada ser completa, coerente e fácil de compreender (problema identificado: a falta de consciencialização da importância da documentação associada ao software) (Chapin, et al., 2001).

A fase de manutenção tem maior duração que as outras fases, e normalmente atinge a proporção de anos. Desta forma também é constatada rotação de pessoas durante esta fase. A rotação nesta fase é compreensiva visto que o programador tem um trabalho repetitivo e monótono, e na maioria das vezes com pouco valor acrescentado.

A passagem da fase de desenvolvimento para a fase de manutenção é um dos pontos cruciais: quem desenvolve, não é quem mantém; com a agravante que quem desenvolve não faz documentação suficiente para a ser realizada a manutenção. Um dos problemas identificados na área de MS, na comunidade científica, é o esforço desenvolvido pelos novos programadores para a compreensão do código herdado.

3.1.2. Processo ineficaz e ineficiente

Tendo encontrado o motivo base para os problemas descritos na comunidade científica, não podemos deixar de ir a um nível superior: ao nível do processo. Na maioria das empresas não existe um processo de passagem entre as actividades, nem um processo standard para realizar manutenção. Assim o processo de manutenção é caracterizado como ineficaz e ineficiente.

Quando um processo é ineficiente, significa que apesar do processo estar a funcionar, não está a produzir os resultados óptimos. Isto é o que acontece na empresa onde estive colocada durante 10 meses, onde raramente são cumpridos os objectivos pretendidos pelo negócio. Como se pode através do gráfico dos pedidos de Manutenção Aplicacional (Anexo 4), existe um número elevado de pedidos que não são entregues no prazo estabelecido. Contudo, são sempre entregues ao utilizador final, seguindo um processo específico, desde a definição de requisitos aos testes realizados pelos utilizadores finais. Está também relatado no capítulo 2 onde se descreve que apesar de existir um processo, os resultados da sua execução ficam aquém das expectativas.

Quando um processo é ineficaz, significa que não produz efeito. Ou seja, o processo de manutenção de software é inexistente, ou a forma como está institucionalizado não resolve o procedimento que está em falta. Em várias empresas, não existe um processo para realizar a manutenção do software no geral. Isto verifica-se pois as empresas preferem desenvolver à medida que vão precisando (não existindo um inteiramento das versões produzidas ao longo do tempo) ou porque não têm consciência que a longo prazo, ou evoluem ou os seus programas ficam inadequados, não podendo responder aos desafios do mercado (em constante mudança). Não vêem a necessidade de possuir a habilidade de alterar o software que hoje responde às necessidades do negócio, mas amanha pode já não responder.

A maioria do software é criado para objectivos concretos/específicos e, normalmente, não são tidos em atenção, ou seja não é dada importância, aos assuntos e questões genéricas, tais como a manutenção do produto, a durabilidade ou mesmo a preparação para a retirada do produto.

3.2. Definição

De um modo geral, considerei que os problemas podem ser pensados em conjuntos distintos dependendo da forma como têm impacto sobre o processo de manutenção de software.

Desta forma, existem três níveis em que podemos encarar os problemas. O primeiro é ao nível da Empresa, em que o problema afecta toda a organização, causando impacto nos seus processos de negócio, orçamentos, e na gestão da própria empresa. Estes problemas são enunciados de uma forma ao mais alto nível e precisam de uma estratégia de negócio para ser modificados.

O segundo é ao nível do processo da manutenção de software, e engloba os problemas relacionados com os intervenientes sobre os quais o processo de MS tem impacto: utilizadores finais, analistas/programadores.

O terceiro é ao nível de requisitos de sistema, quer de hardware quer de software.

De um modo geral, os problemas que se fazem sentir na realidade empresarial são os seguintes:

- Inabilidade de mudar fácil e rapidamente o software;
- Não responder às necessidades do negócio/utilizadores;
- Não existe um processo considerado standard pelas organizações;
- Fraca qualidade da documentação;
- Não são antecipadas mudanças na fase de desenvolvimento;
- Grande número de recursos humanos desempenha um trabalho repetitivo e monótono.

Ao nível do âmbito da manutenção de software, o processo é operado por 2 intervenientes: os utilizadores, responsáveis pelo desencadeamento do procedimento; os programadores, responsáveis pela execução dos pedidos dos utilizadores. Nestes moldes, podemos também especificar factores que levam o procedimento de manutenção a falhar.

De seguida, estão identificados os factores que potenciam essas mesmas falhas:

- Competências/Conhecimento do utilizador. Este conhecimento é o essencial do programa sobre o qual opera;
- Falta de compreensão do utilizador;
- Treino inadequado;
- Eficiência/eficácia do programador, se quem efectua a manutenção tem produtividade, motivação e *skills*;
- Falta de garantia que o Programador terá: as especificações de desenho adequadas; qualidade na programação; qualidade na documentação e disponibilidade;

Quanto ao nível de requisitos do sistema, os problemas encontrados são:

- Qualidade do produto, não existem modo (métricas) com que se possa avaliar a qualidade do software que sofreu alterações.
- Requisitos de Hardware, o hardware tem de permitir que determinadas alterações sejam possíveis de ser executadas tendo em conta o hardware que está a ser utilizado efectivamente.
- Fiabilidade/disponibilidade do sistema, garantir os níveis de serviços acordados entre as partes, após entrada em produção da versão passou pelo processo de manutenção.
- Requisitos de armazenamento e tempo de processamento, de fiabilidade do sistema de hardware e software, integridade dos dados.

Dos factores enunciados, o conhecimento do utilizador ocupa 60% do total de problemas existentes, dando assim prova da importância à relação com o utilizador para a determinação do sucesso ou fracasso de um sistema. Os problemas de eficiência do programador e qualidade de produto forma maiores para sistemas antigo ou de grande dimensão e onde foi mais empenhado esforço para a manutenção correctiva. Ambiente de processamento de dados de grande escala estavam significativamente associados a grandes problemas (problemas maiores) ou eficácia do programador, relatavam qualquer um dos outros factores enunciados.

A qualidade do produto foi visto como o menor dos problemas, quando determinadas técnicas de produtividade eram utilizadas durante o desenvolvimento do sistema

Em resumo, no âmbito dos factores acima descritos, a questão que colocamos é:

- Como optimizar o processo de Manutenção de Software?

A questão prende-se com o facto de que já existem documentos que afirmam que a manutenção não é um problema, mas sim uma solução (Glass, et al., 2006). Desta forma existe um interesse crescente para optimizar a forma como a manutenção está a ser encarada quer pelas empresas quer pela comunidade científica.

4. Proposta

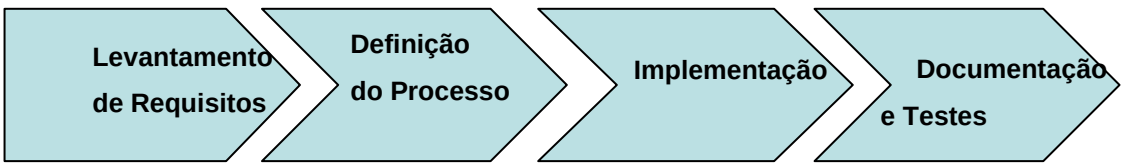
O caso de estudo desta tese é o departamento de Sistemas de Informação (DSI) de uma empresa que opera em full-*outsourcing*. A proposta é implementar um dos processos do ITIL, num dos processos do DSI. O processo de manutenção aplicacional é o processo que será alvo de estudo, sendo o seu correspondente no ITIL o processo de Gestão de Alterações.

A solução proposta provém do estudo realizado nos primeiros capítulos desta tese. Tendo em conta o problema central da tese as três soluções apresentadas serão adequadas. O ideal seria conseguir conciliar as três soluções, podendo desta forma tirar proveito das vantagens que cada uma delas oferece.

Desta forma a proposta para a optimização do processo de MS baseia-se em:

- Aproveitar o facto do processo estar a ser executado em *outsourcing*, visto ser uma das principais razões que leva à optimização da manutenção;
- Redefinir um processo para a manutenção aplicacional tendo como base um processo da framework ITIL;
- Definir um processo de documentação (para a fase de desenvolvimento e para a fase de manutenção).

Inicialmente, para optimizar o processo de manutenção de software, é necessário compreender a situação actual da empresa. Para tal é necessário: efectuar um levantamento de requisitos; desenhar um novo processo adequado à empresa tendo por base o processo de Gestão de Alterações do ITIL; e por fim implementar esse novo processo e verificar a sua eficácia face ao processo anterior. Através dessa verificação ficará provado que é possível implementar o ITIL numa empresa de full-*outsourcing*.



Fig, 10 – Passos para a elaboração da proposta.

Uma vez enunciadas as soluções propostas apresento de seguida o motivo, nomeadamente as vantagens que cada uma das soluções por si poderá trazer à Manutenção de Software.

4.1. Porquê a aposta no *Outsourcing*?

O processo de MS que irei abordar no caso de estudo, encontra-se me outsourcing. De seguida passo a enumerar as razões que levam as empresas a optar por esta solução.

A aposta no *outsourcing* prende-se especialmente com 3 razões:

- Mão-de-obra especializada. Recorrendo ao *outsourcing* é possível assegurar determinados serviços com mão-de-obra dedicada, que lida eficazmente com os nossos sistemas e utilizando todas as suas potencialidades, mantendo sempre o nível de serviço acordado.
- Tecnologias de ponta. Uma vez que estamos a trabalhar em *outsourcing* podemos estar sempre a par das últimas tecnologias/inovações que entram no mercado e, sendo o caso, trocar de parceiro para determinado serviço.
- Redução de custos. A redução de custos acaba por ser automática, visto que a passagem de um serviço para *outsourcing* implica a redução de recursos humanos na empresa.

A criação de economias de escala, apesar de não ser directamente uma vantagem para a MS, vai permitir um leque maior de escola para as empresas, e a consequente rivalidade entre si ira ter repercussões a nível dos custos imputados nos seus serviços. Transmitindo-se desta forma, numa pequena redução de custos e aumento de eficiência para as empresas que irão recorrer ao *outsourcing*.

4.2. Porquê a aposta no ITIL?

A aposta no ITIL tem duas razões: a primeira prende-se com o facto das suas directrizes preverem a abordagem de processos integrados para a entrega de serviços, planeamento e implementação da gestão de serviços, controle de processos, entre outros; a segunda razão é relacionada com o tópico da qualidade. Consolidada no fim do ano 2005, a norma ISO 20000 carrega o status de primeiro padrão global para gestão de serviços de TI e é baseada nas melhores práticas da biblioteca do ITIL.

4.3. Porquê a aposta na documentação?

Existem inúmeras razões para apostar numa boa documentação, nomeadamente porque:

- Permite poupar tempo de conhecimento;
- Melhora a compreensão das escolhas feitas pelo programador;
- Um vez existindo leis, como o SOX, que obrigam à existência de documentação, é normal que exista uma tendência para a existência de uma norma semelhante, ao nível interno das empresas, para o desenvolvimento e manutenção de software.

5. Caso de estudo

Durante o último ano estive envolvida no Departamento de Sistemas de Informação (DSI) de uma grande empresa portuguesa que opera em *outsourcing* total. Este envolvimento teve a duração de 10 meses, durante os quais pude conhecer a realidade da mesma, angariando conhecimento e experiência através da realização de três projectos.

O primeiro consistia no desenho de processos e é abaixo indicado como Desenho de Processos DSI, o segundo era a optimização de um dos processos desenhados anteriormente e está abaixo indicado como Processo Manutenção Aplicacional (MA), e por último a apresentação do processo em causa através de um protótipo, abaixo explicado como Protótipo.

5.1. Desenho de Processos DSI

Criada no final dos anos 70, a empresa em estudo é um operador ibérico de soluções energéticas que desenvolve as suas actividades nas áreas de Produção, Comercialização e Distribuição de Electricidade e Comercialização e Distribuição de Gás. Esta empresa teve um grande crescimento nos últimos anos apoiando-se numa estratégia de crescimento das áreas de Gestão, Finanças, Engenharia e Marketing.

Uma das preocupações desta empresa é a sua cotação na bolsa de Nova Iorque. Motivada por escândalos financeiros corporativos (Enron, Tyco International, Peregrine Systems e WorldCom) foi criada uma lei, a lei Sarbanes-Oxley, que deve ser seguida por todos os interessados em manter a sua posição na bolsa de Nova Iorque.

Lei Sarbanes-Oxley

Esta lei foi assinada em 30 de Julho de 2002, pelos senadores Paul Sarbanes (Democrata de Maryland) e Michael Oxley (Republicano de Ohio).

Motivada por escândalos financeiros corporativos, entre eles o caso da Enron, que acabou por afectar drasticamente a empresa de auditoria Arthur Andersen, esta lei foi redigida com o objectivo de evitar o esvaziamento dos investimentos financeiros, e a fuga dos investidores, causada pela aparente insegurança quanto à governação adequada das empresas.

Esta lei, denominada lei Sarbanes-Oxley, foi apelidada de Sarbox ou ainda de SOX. No seu conjunto procura garantir a criação de mecanismos de auditoria e segurança fiáveis nas empresas, incluindo ainda regras para a criação de comités encarregados de supervisionar as suas actividades e operações, de modo a diminuir os riscos dos negócios, evitar a ocorrência de fraudes, ou dispor de meios para identificar quando elas ocorram, garantindo a transparência na gestão das empresas.

Alguns requisitos desta lei passam por tópicos como:

1. Controlar a criação, edição e controlo de versões dos documentos num ambiente de acordo com os padrões ISO;

2. Registar os riscos associados aos processos de negócio e armazenar os desenhos de processo;

3. Utilizar ferramentas como Word e Excel para criação e alteração de documentos;

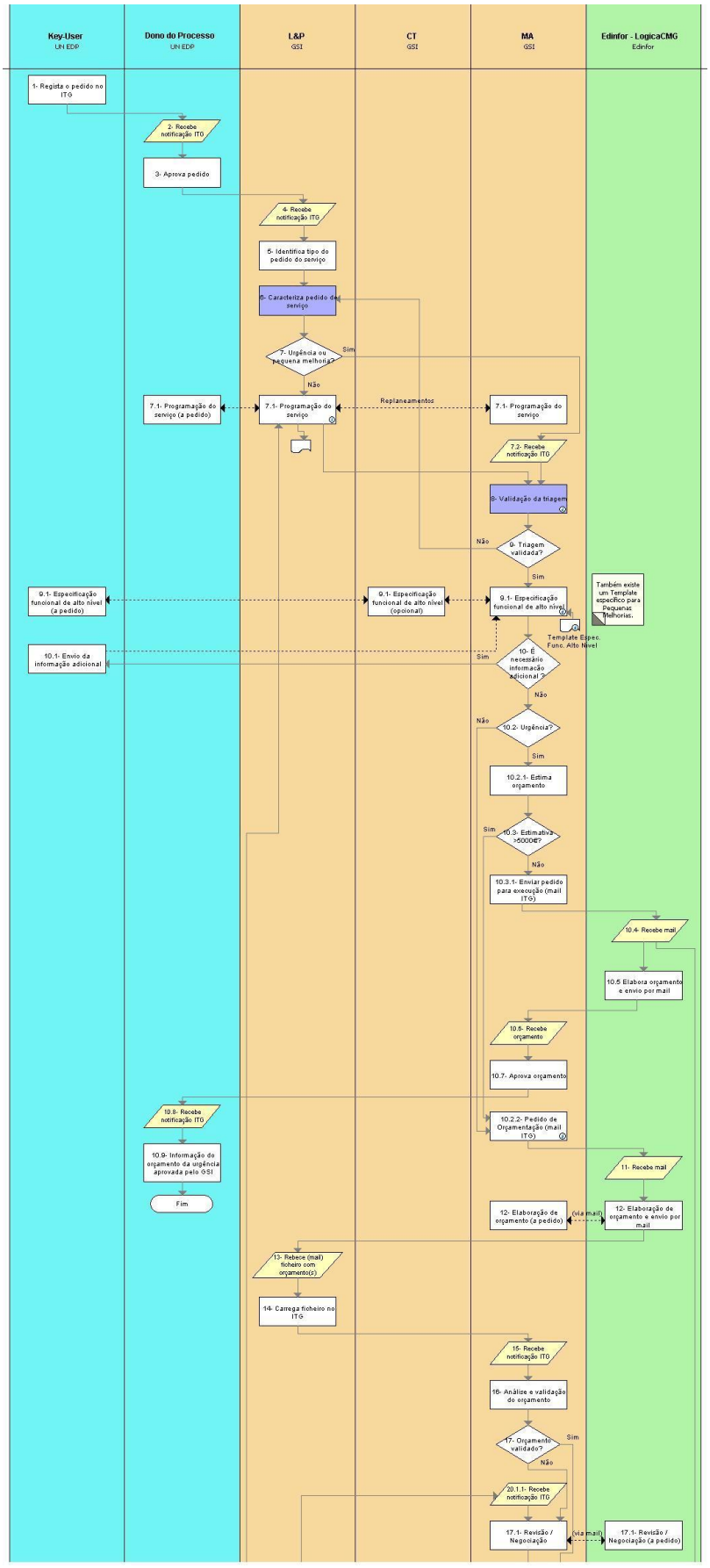
4. Digitalizar e armazenar todos os documentos existentes em papel.

Posto isto, o início da minha participação na DSI foi através do desenho dos seus processos de negócio. Para a realização desta actividade foi necessário compreender a cultura da empresa, a sua estratégia e, mais importante, as suas unidades e áreas de negócio. Sendo reduzida a documentação para este efeito, para além da sua leitura, foi necessário estar com as pessoas de cada uma das unidades de negócio, de modo a compreender claramente o objectivo dos passos de cada processo a ser desenhado.

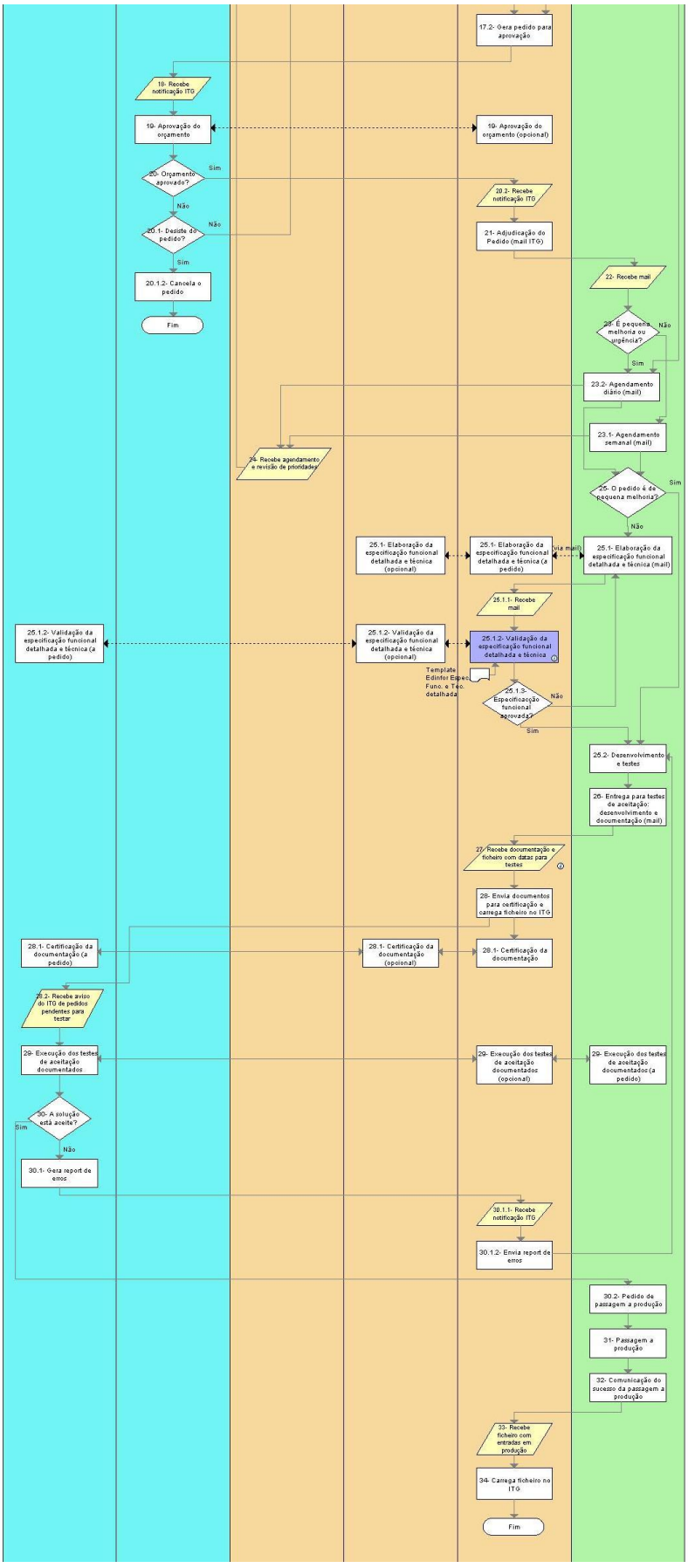
A DSI é composta por várias áreas, e essas por sub-áreas. Por exemplo, no caso da MA (Manutenção Aplicacional), existem 3 vertentes: Correctiva, Preventiva e Evolutiva. De forma a ter um conjunto de processos coerentes, quer os que são dentro da mesma área quer os que atravessam duas ou mais áreas, é necessário realizar este desenho de uma forma interactiva e incremental. Da mesma forma, deve haver um consenso entre todos os chefes de área, sendo também necessária a aprovação do *owner* do processo (normalmente um dos chefes de área).

O resultado deste projecto foi um conjunto de processos de negócio modelados em conformidade com as políticas da empresa e de acordo com a medida SOX. O número total de processos desenhados é de 47, encontrando-se em anexo (Anexo 3) os que se relacionam com o tema do estudo. Cada processo está associado a um conjunto de documentação que descreve e caracteriza as suas actividades.

Nas figuras 11 e 12 está descrito o processo mais relevante para este trabalho, sob a notação utilizada pela empresa – QPR. Esta notação encontra-se descrita em mais detalhe na secção Tecnologia QPR, no final do capítulo.



Fig, 11 – Processo antigo MA – 1ª parte



Fig, 12 - Processo antigo de MA – 2ª parte

5.1.1. Tecnologia QPR (QPR, 1991)

A ferramenta utilizada para a modelação dos processos em questão é denominada QPR. O QPR é uma ferramenta interactiva para planeamento, implementação, comunicação e aperfeiçoamento de processos de negócio. Esta permite às organizações modelar, monitorizar, gerir e analisar os seus processos de negócio, atingindo assim diversos benefícios na gestão de processos, tal como a obtenção de informação em tempo real.

Sendo uma ferramenta multi-utilizador, a tarefa de modelação dos processos teve dois tipos de interactividade. A primeira é a publicação, para todos os intervenientes do processo poderem aceder ao mesmo. O segundo é o envio de mensagens através de um *highlight* na actividade em questão. Estas características permitem ao QPR ser uma ferramenta colaborativa, disponível a todos os elementos da organização através de uma *intranet*.

5.2. Processo Manutenção Aplicacional

Como se pode observar nas figuras 11 e 12, todo o processo de Manutenção Aplicacional assenta sobre uma política de *outsourcing*, e a gestão do serviço contratado é da responsabilidade da área em questão.

A gestão é feita da seguinte forma: existe um contrato, em que são acordados os níveis de serviço (SLAs). Neste caso, o DSI pode usufruir de uma quantidade determinada de horas de serviço do *outsourcer* por mês. Cada pedido efectuado ao *outsourcer,* pela DSI, necessita de um acordo quanto ao número de horas necessárias para o desempenho desse mesmo pedido, funcionando como orçamento.

O DSI trabalha em regime de *full-outsourcing* e tem contratos com duração de vários anos, tornando possível pré acordar o orçamento para os pedidos executados periodicamente. Esta situação leva à criação de um processo que agilize este tipo de pedidos, denominados Pedidos Tipificados.

Uma vez estudado o processo anterior, está representado o levantamento de requisitos para a concretização do segundo projecto: a optimização do processo de manutenção.

Para a realização desta fase foram seguidos passos:

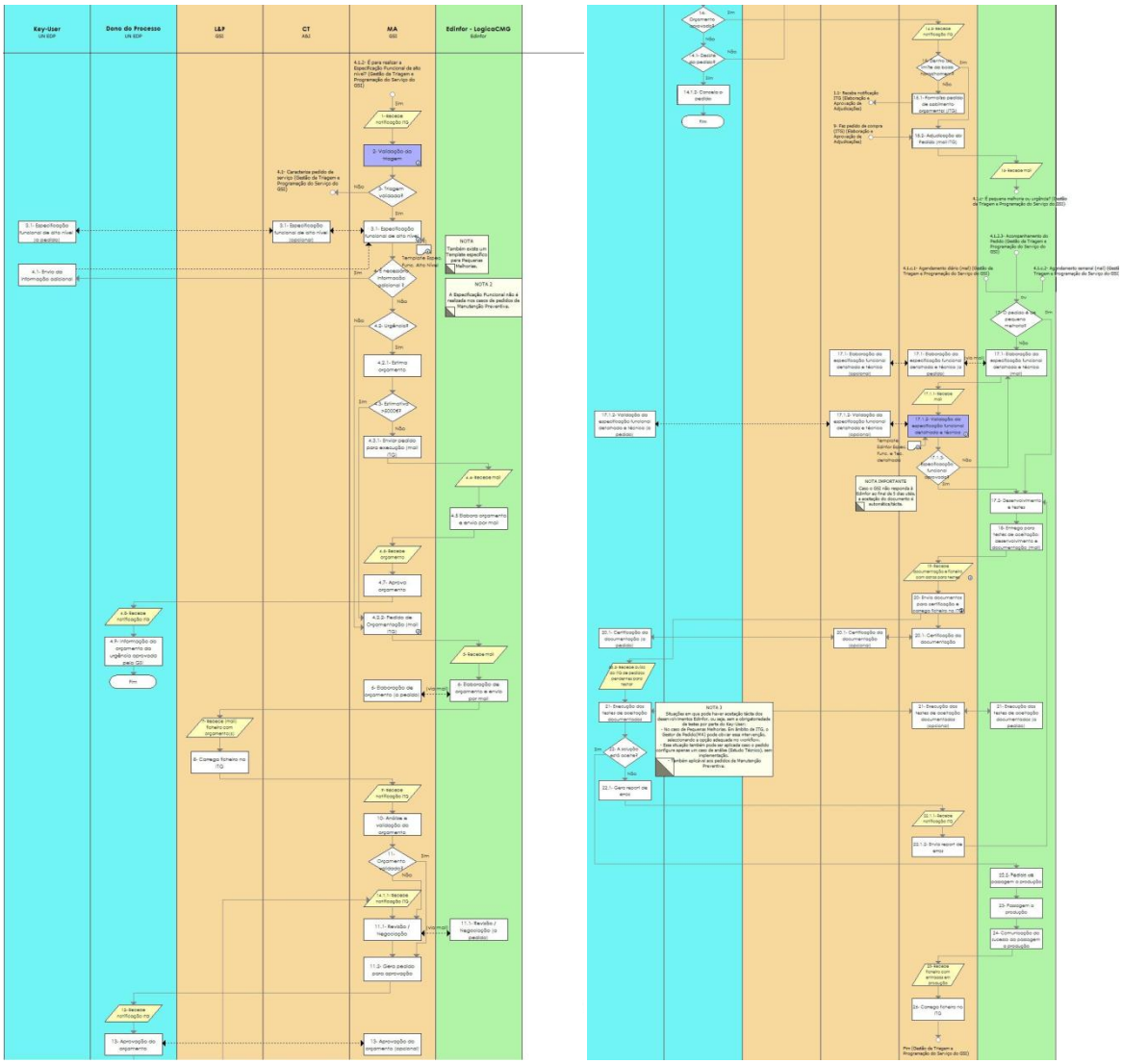
- Identificação os processos relacionados com a manutenção de software e simultaneamente com a Gestão de Alterações (ITIL). Para tal, foi elaborado um mapeamento entre os processos existentes no DSI e os processos existentes no ITIL (Tabela 2).

Mapeamento ITIL (processo dominante)	**Processos DSI (documentados no QPR)**
Incident Management	Escalamento de Incidentes Tratamento de Incidentes e Pedidos Pendentes (backlog)
Problem Management	Gestão de Problemas (Prog. Qualidade) Gestão de Reclamações GSI
Change e Release Management	Gestão da Triagem e Programação do Serviço do GSI Pedido de Revisão de Prioridades Gestão de Ambientes Aplicacionais (CPD) Gestão de Pedidos de Infra-estruturas Gestão de Pedidos de Manutenção Evolutiva Alteração de Processo GSI Existente Criação de Novo Processo GSI Supressão de Processo GSI Existente Gestão de Projectos GSI Alteração de Âmbito de Projecto Gestão de Propostas de Manutenção Preventiva Gestão Documental (inclui Portfolio de Aplicações) Alteração da Criticidade das Aplicações EDP Pedidos de Acesso de Infra-estruturas e Aplicações Pedidos de Acessos Aplicacionais (AS-IS) Fluxo SAP HR - Criação, Alteração ou Supressão de Acessos Pedidos de Acessos Internacionais (Infra-estruturas e Aplicações) Pedidos de Indisponibilidade (CPD) Pedidos de Indisponibilidade (Servidores Distribuídos e redes/WAN)
Mapeamento ITIL (processo dominante)	**Processos DSI (documentados no QPR)**
Service Level Management	Comité de Acompanhamento e Planeamento - Processo Controlo dos Níveis de Serviço - Relatório de Serviço Controlo dos Níveis de Serviço - Relatório Global de Serviço
Financial Management for IT Services	Acompanhamento e Reporting Orçamental Cross Checking dos dados SAP c/ EDP Valor Formalização de Critérios de Chargeback - Contratos Formalização de Critérios de Chargeback - Projectos Formalização de Novos Critérios de Chargeback (On-going) Formalização do Dossier Anual de Chargeback Auditorias Ciclo Orçamental Orçamento - Investimentos em Projectos (rúbrica CAPEX)
Security	Revisão periódica de contas e perfis de utilizadores - Aplicações
Outros (não ITIL)	
Provisioning	Pedido de Proposta (RFP e Compras) em Projectos Avaliação de propostas Renovação de Contratos de Licenciamento Validação e Facturação de Consumos Elaboração e Aprovação de Adjudicações
Gestão RH	Admissões de RH no GSI
Documentos Diversos	Procedimentos diversos Gestão de Processos GSI Relacionamento Edinfor Gestão do Plano Estratégico de Sistemas (PESI) Renovação Tecnológica de Postos de Trabalho e Periféricos

Tabela 2 – Mapeamento Processos DSI - ITIL

- Eliminar passos similares;
- Inserir pontos de documentação;
  - A inserção de pontos de documentação é a explicação detalhada de cada actividade do diagrama do processo. Para cada actividade podem existir *templates*, standards, ou outro tipo de documentos que têm de ser consultados para uma execução mais rápida e melhor de determinada actividade.
- Fundir os princípios do processo de Gestão de Alterações, sugerido no ITIL, no processo existente no DSI.

Foi então sugerido o processo representado na fig. 13:



Fig, 13- Processo Novo de MA

Uma vez desenhado o processo, optou-se por dar como exemplo um pedido de menor complexidade da Gestão de Alterações do ITIL. No contexto do DSI, este processo é denominado Processo de Tipificados.

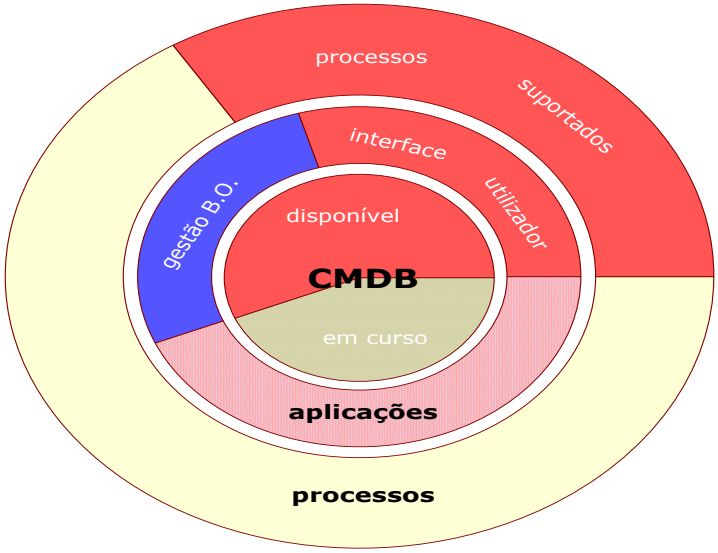
Estes pedidos menores apresentam uma complexidade baixa e são realizados num curto espaço de tempo. Desta forma, quanto mais pedidos se puderem categorizar como menores, menos pedidos terão de ser tratados pelo processo normal.

O meu trabalho passou por enumerar os pedidos mais efectuados em cada área do DSI, criando assim um catálogo de pedidos.

5.3. Protótipo

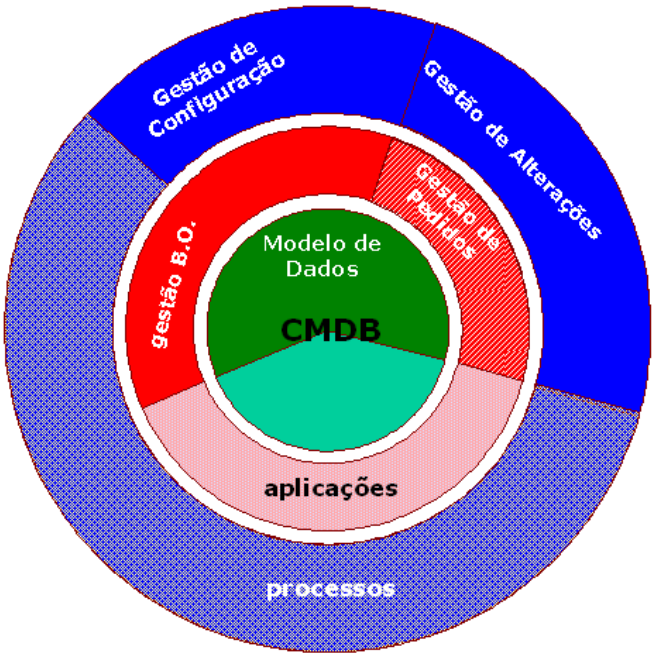
Durante o decorrer dos meus projectos, estava a ser desenvolvido, no DSI, um software de acordo com as boas práticas do ITIL. O ITIL está dividido em 13 processos, os quais assentam sobre o conceito de uma base de dados global: a CMDB. Todos os processos acedem à informação da CMDB, enquanto que apenas alguns processos têm autonomia para inserir e alterar os seus dados.

No DSI estava a ser desenvolvida uma CMDB, o *backoffice* da CMDB e ainda um Gestor de Configurações. Como se pode ver na figura, a CMDB é o núcleo de informação e pode ser sempre acrescentado um novo processo (representado como uma camada em cima do núcleo).



Fig, 14- Arquitectura em camadas da aplicação desenvolvida no DSI -

O trabalho realizado no DSI representa a vertente da Gestão de Alterações (*Change Management*, nomenclatura ITIL). O protótipo criado é inserido no software existente da seguinte forma:



Fig, 15 - Arquitectura em camadas com o módulo Gestor de Alterações.

O sistema apresentado é composto por camadas independentes entre si, tornando-o de fácil alteração e manutenção.

5.3.1. Tecnologia *OutSystems*

A tecnologia utilizada para realizar o protótipo foi o *OutSystems*. Esta escolha foi determinada pelo facto de querer integrar o protótipo com o núcleo aplicacional já existente. Manter a mesma tecnologia era a escolha apropriada.

De qualquer forma, esta linguagem apresenta-se como uma boa escolha, visto ter as seguintes características:

- Rápido desenvolvimento;
- Primitivas gráficas;
- Gerador de documentação.

Uma vez desenvolvido o processo, foi necessário fazer uma implementação num protótipo de modo a demonstrar a sua utilidade e a sua eficácia aos vários membros do DSI, de modo a que a mudança de um processo para o outro fosse feita com a segurança de ter os utilizadores satisfeitos.

A implementação do protótipo está associada a dois documentos:

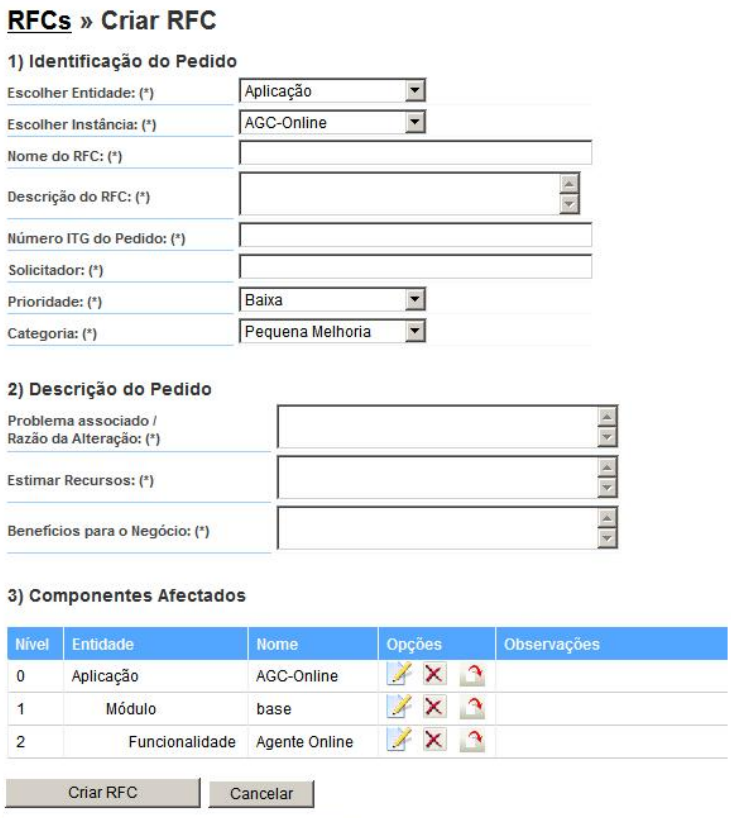
- Especificação Funcional – onde estão descritas em detalhe as funcionalidades do protótipo (Anexo 1);
- Especificação Técnica – onde está descrito o modelo de dados (estruturas e entidades), as funções e os seus parâmetros e ainda o diagrama de contexto do protótipo face às aplicações existentes (Anexo 2).

Os documentos acima descritos encontram-se anexados na sua totalidade.

5.3.2. Implementação do Gestor de Alterações – Especificação Funcional

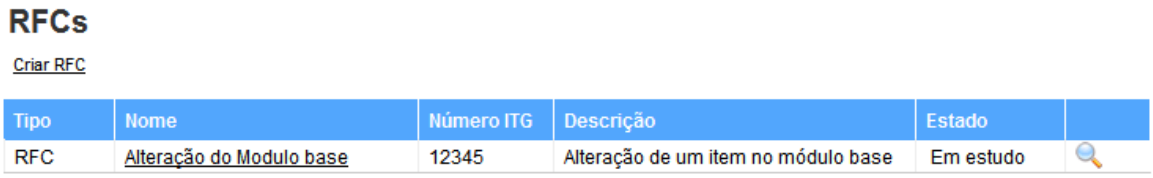
O Gestor de Alterações disponibiliza ao utilizador o seguinte conjunto de funcionalidades:

- Criar Pedido de Alteração (denominado RFC);
  - Esta funcionalidade permite ao utilizador criar o RFC baseando-se em dados existentes na CMDB.



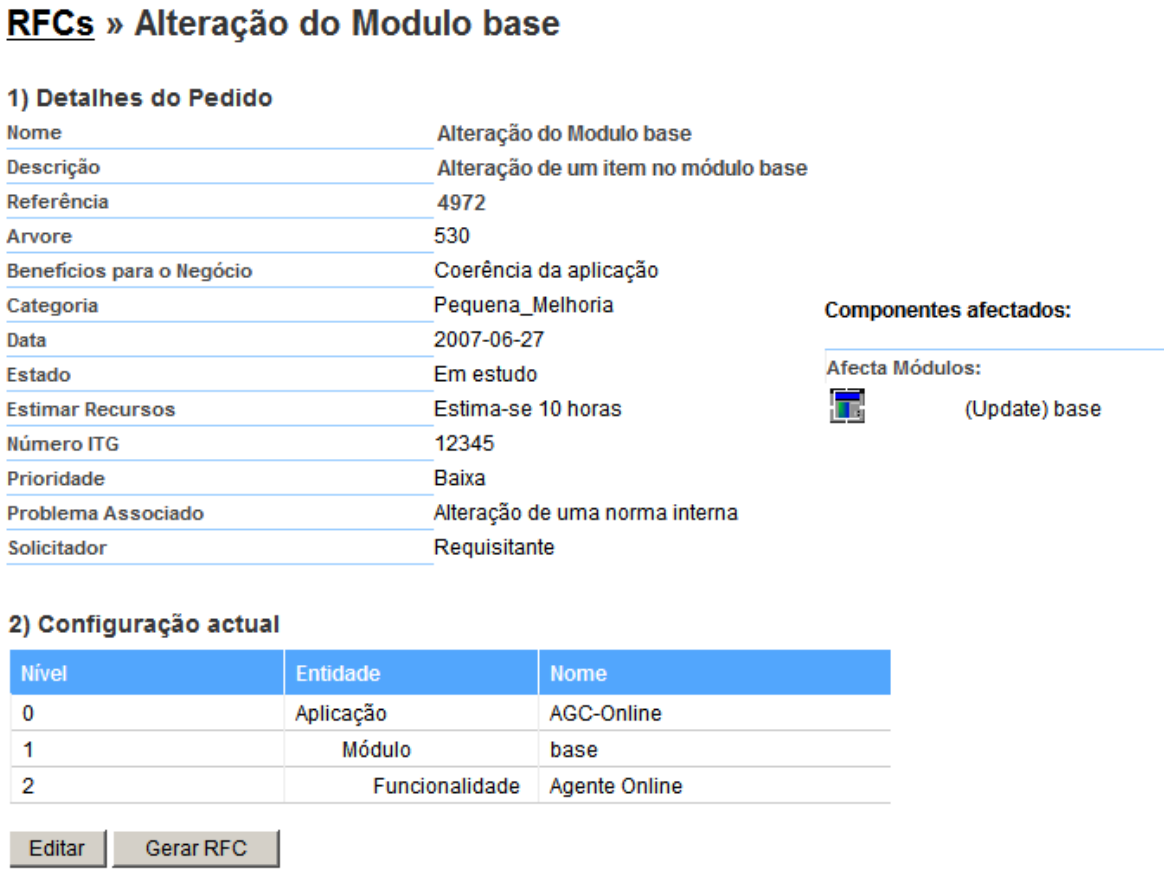
Fig, 16 - Interface Criar RFC

- Listar RFCs;
  - Esta funcionalidade é apresentada no ecrã inicial do sistema. Lista todos os RFCs criados, independentemente do seu estado.



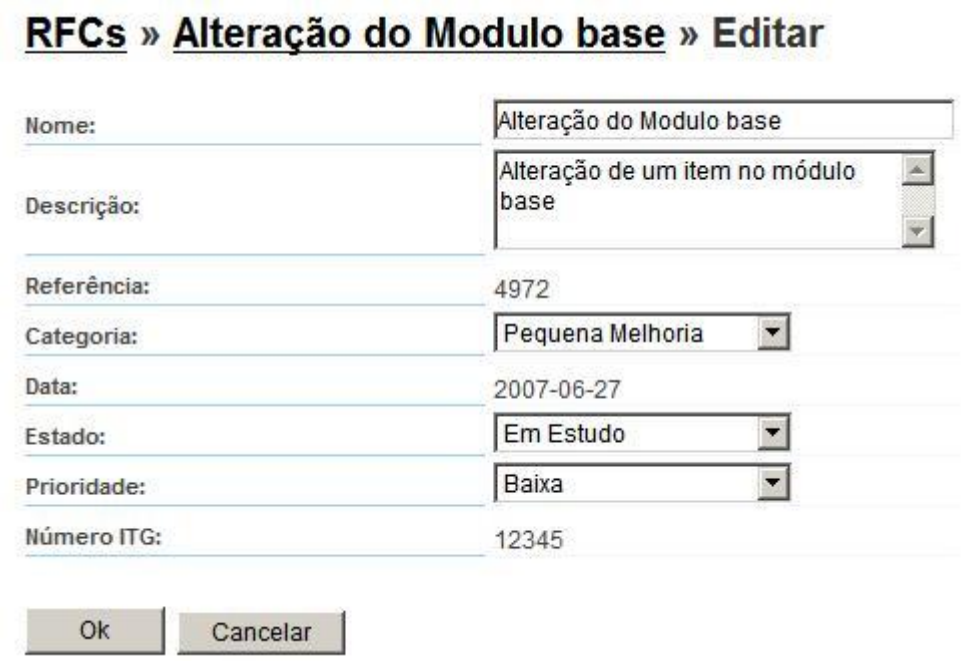
Fig, 17 – Interface Listar RFC

- Detalhes RFC;
  - Esta funcionalidade permite ver os detalhes associados ao RFC, escolhido pelo utilizador. Esta funcionalidade pode ser acedida através do ecrã que contém a listagem dos RFCs, ou seja, do ecrã principal.



Fig, 18 – Interface Detalhes RFC

- Actualizar RFC.
  - Esta funcionalidade permite ao utilizador editar as propriedades do RFC, definidas quando o mesmo foi criado. A nova informação é guardada e fica logo disponível para consulta. Esta funcionalidade está disponível a partir do ecrã 'Detalhes RFC'. O campo Estado é o campo de maior relevo da edição do RFC, por desencadear outras funcionalidades no futuro, tais como adicionar novos documentos ao RFC (EFN e Orçamento).



Fig, 19 - Interface Actualizar RFC

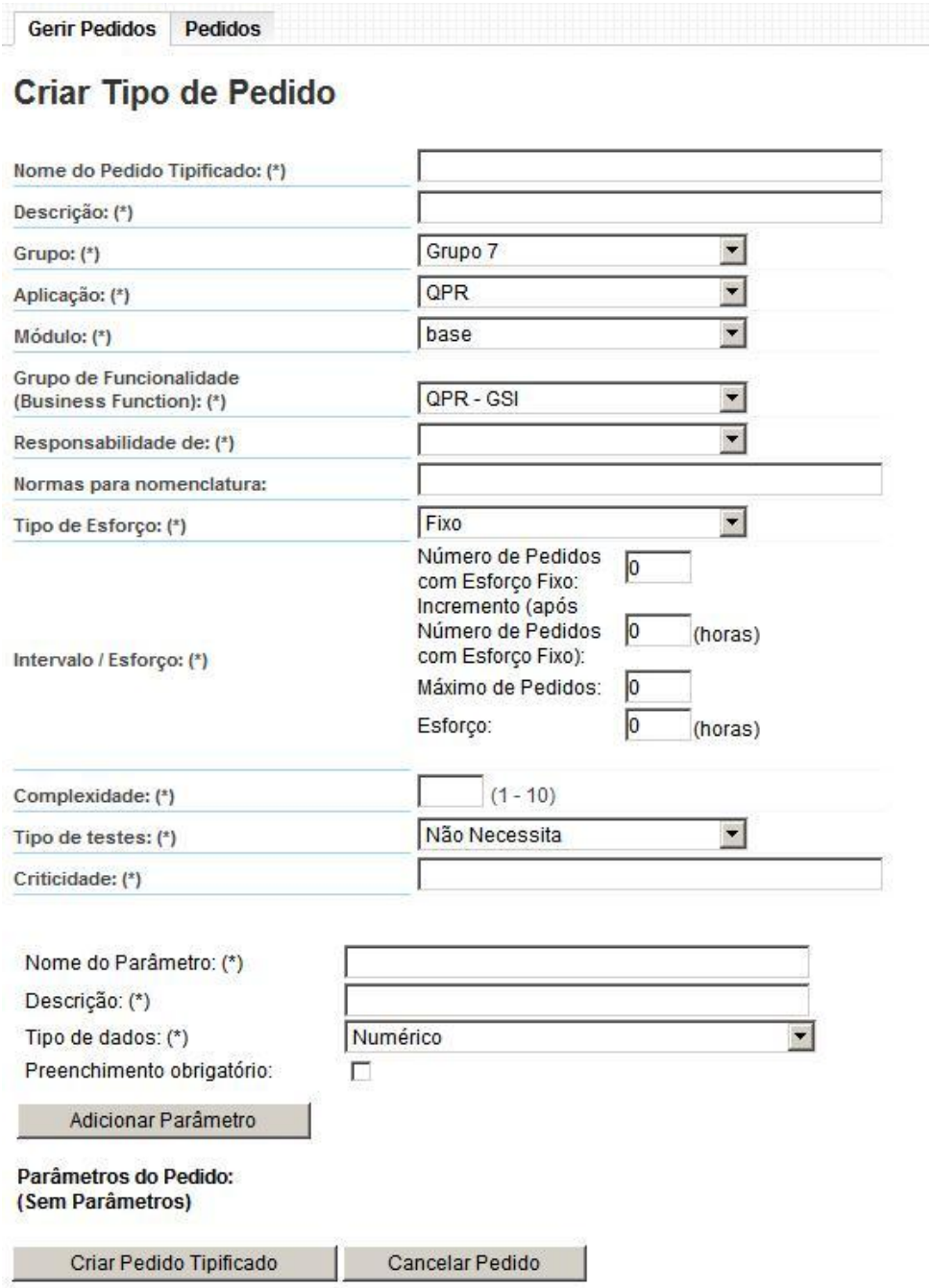
5.3.3. Implementação do Gestor de Pedidos Tipificados – Especificação Funcional

Após a implementação do Gestor de Alterações, foi implementado o sub-processo referente aos pedidos de baixa complexidade, já referidos no início do capítulo - os pedidos tipificados.

A implementação deste protótipo tinha como objectivo funcionar como processo piloto para o Gestor de Alterações.

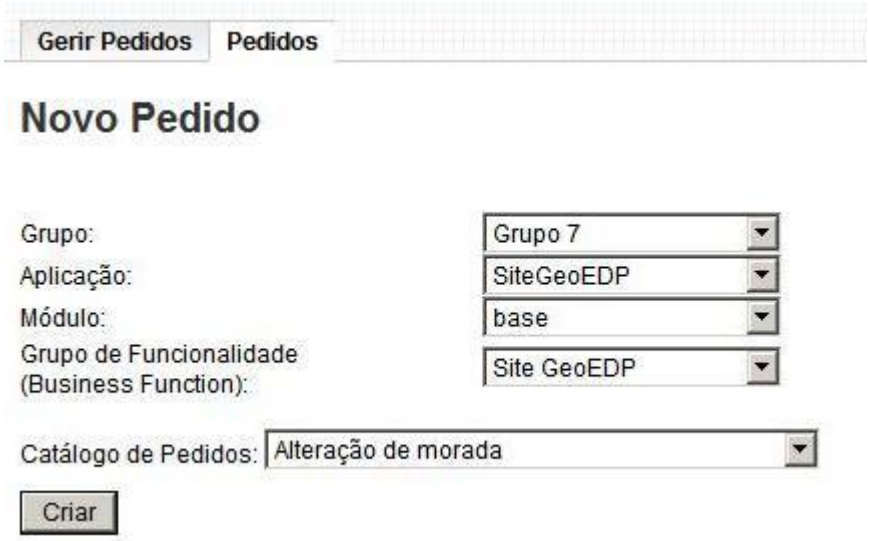
O Gestor de Pedidos Tipificados disponibiliza ao utilizador o seguinte conjunto de funcionalidades:

- Criar Tipo de Pedido:
- Esta funcionalidade permite ao utilizador criar um novo tipo de pedido tipificado. Após a sua criação, o tipo vai passar a estar disponível para utilização pelos DP/KY, através do Catálogo de Pedidos;



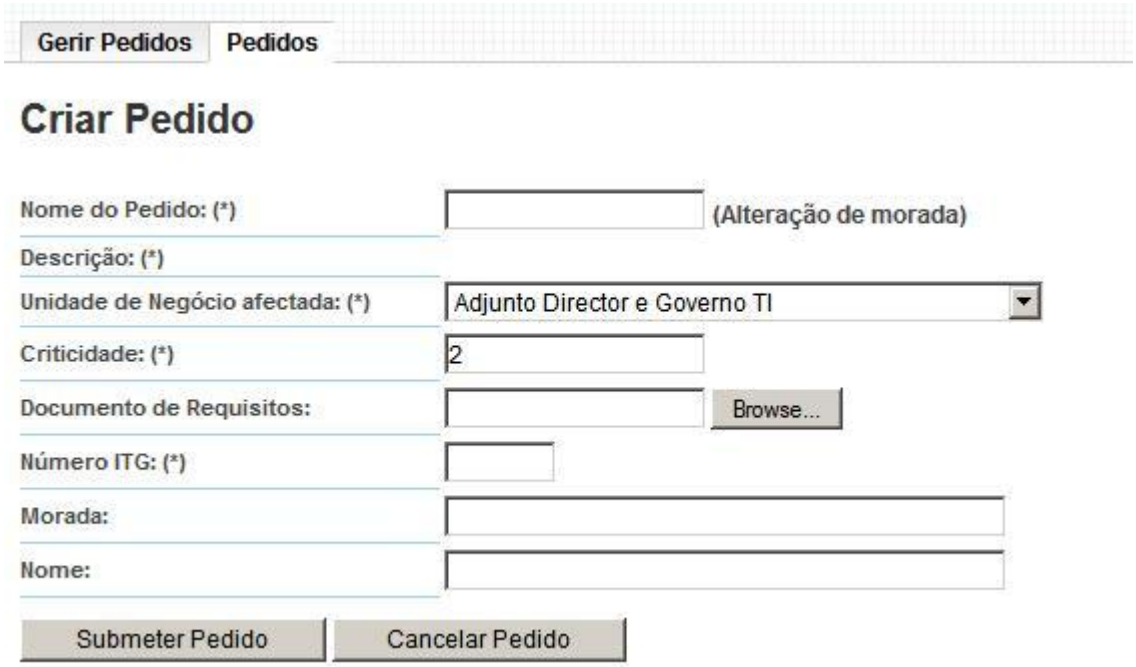
Fig, 20 - Interface Cria Tipo de Pedido

- Criar Novo Pedido;
    - Esta funcionalidade permite ao utilizador criar um novo pedido tipificado. Existe um Catálogo de Pedidos, onde o utilizador pode escolher qual o pedido que pretende instanciar.



Fig, 21 - Interface 1 Criar Novo Pedido

- Após premir o botão “Criar”, o utilizador deve preencher os campos do pedido apresentados no ecrã e premir o botão “Submeter Pedido”.

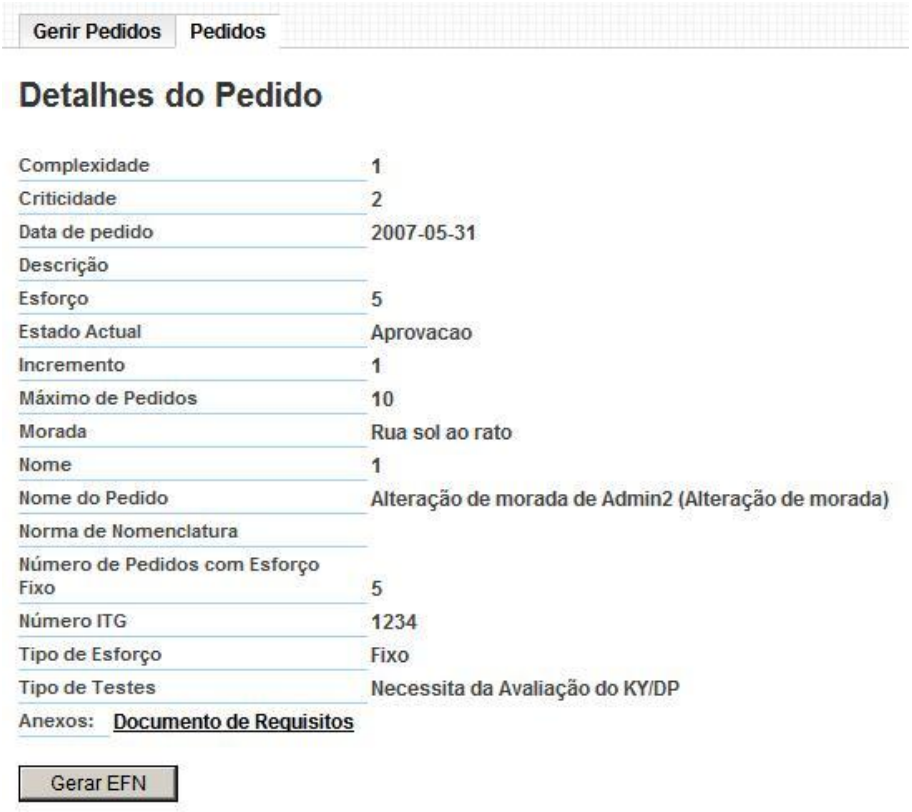


Fig, 22 - Interface 2 Criar Novo Pedido

- Pesquisar Pedido;
    - Pesquisar Pedido – Esta funcionalidade permite ao utilizador pesquisar um pedido através de vários critérios de pesquisa. Esses critérios estão associados a todos os pedidos, como por exemplo o NumITG.

Fig, 23 - Interface Pesquisar Pedido

- Visualizar Pedido – Esta funcionalidade permite ao utilizador visualizar os detalhes de um pedido específico. A selecção do pedido a visualizar implica que foi efectuada em primeiro lugar a pesquisa do pedido.

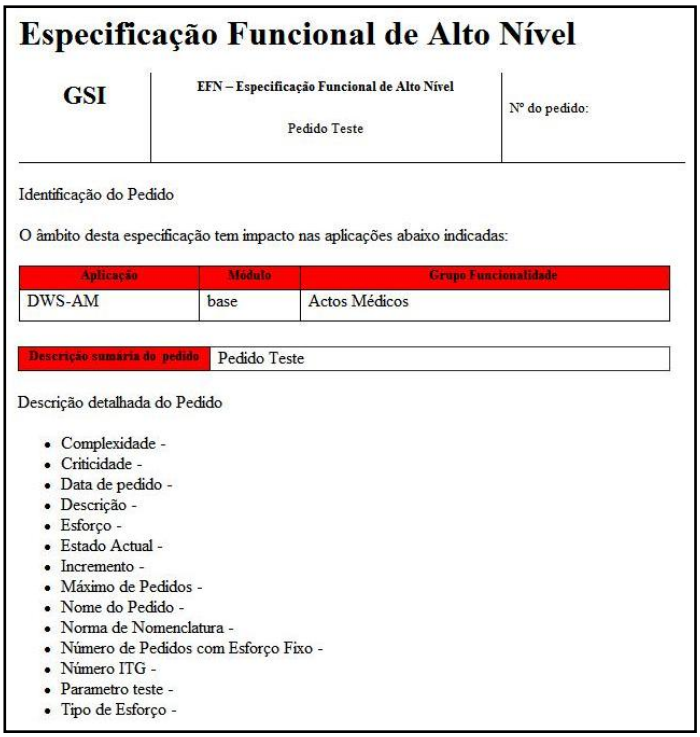


Fig, 24 - Interface Visualiza Pedido

- Registar Utilizador;
    - Esta funcionalidade permite ao utilizador, com o perfil de administrador do sistema, registar utilizadores básicos no sistema, de modo a que a estes seja permitida o acesso à aplicação e às suas restantes funcionalidades.

Fig, 25 - Interface Registar Utilizador

- Gerar Documento do Pedido (EFN).
    - Esta funcionalidade permite ao utilizador gerar a Especificação Funcional de Alto Nível de um determinado pedido. O utilizador pode optar por visualizar ou guardar o documento gerado pelo sistema.



Fig, 26 - Interface Gerar EFN

6. Apresentação e Análise dos Resultados

A análise deste estudo é feita do ponto de vista dos processos, visto ser este o ponto de partida para a optimização da Manutenção de Software.

Foram desenvolvidos dois protótipos com o objectivo de implementar um processo de manutenção que superasse o seu predecessor.

Uma das formas de verificar se uma aplicação cumpre os requisitos para os quais foi criada é desenvolver um determinado conjunto de testes num ambiente semelhante à realidade. Uma vez que a realidade é difícil de replicar, optámos por comparar processos em vez de comparar aplicações.

A comparação de processos consiste no desenho dos mesmos e a utilização de uma ferramenta, denominada “Savvion”, que gera fluxos sobre os processos e mostra qual a aplicação mais ágil segundo determinados critérios. Esses critérios, bem como todo o processo de comparação, estão explicados em detalhe no capítulo 6.1.

O novo processo de manutenção está agora sob influência do ITIL e é executado em *full-outsourcing*. Face ao processo antigo, o novo processo é:

- Ordenado – temos uma sequência de passos com sentido;
- Claro – todos os intervenientes sabem o que fazer em cada actividade;
- Ágil – foram suprimidos passos por não trazer valor acrescido.

6.1. Simulação do processo de negócio

Este capítulo está dividido em três sub-capítulos: explicação da tecnologia utilizada para fazer a comparação, denominada Savvion; apresentação dos processos desenhados; e ainda, a apresentação dos resultados.

6.1.1. Tecnologia Savvion (Savvion, Inc, 1999)

A ferramenta utilizada para simular os processos de negócio denomina-se Savvion.

Savvion é uma ferramenta de modelação, com primitivas semelhantes às primitivas do UML. Uma vez modelado um processo, a ferramenta permite a sua simulação. A simulação é feita recorrendo à alteração de parâmetros como o tempo de cada actividade, o número de instâncias que irá percorrer o processo, o intervalo de inserção entre instâncias, entre outros.

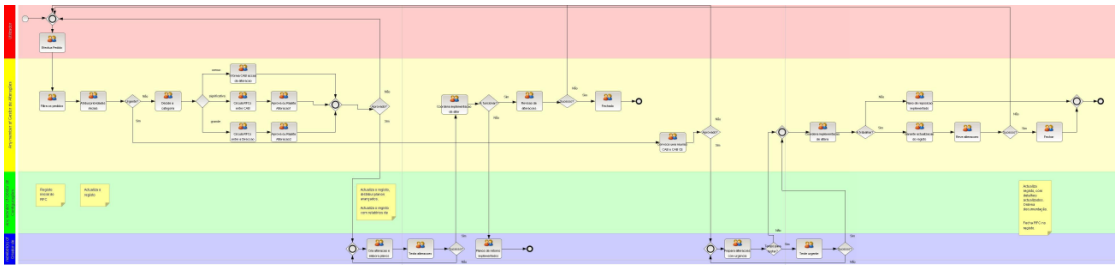
Savvion, ao finalizar uma simulação, produz um relatório com diversos indicadores, que permitem uma análise sobre o processo. Esta análise pode ser em termos de tempo ou custo.

Os relatórios produzidos na ferramenta, bem como os processos desenhados podem ser exportados para outros programas, como o MS Excel, ou guardados noutro formato, como imagem jpeg.

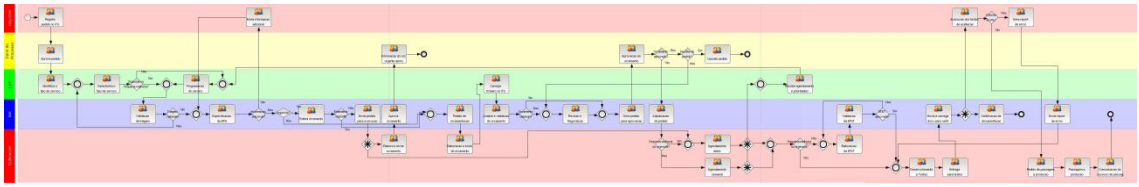
6.1.2. Processos de Negócio

Os processos de negócio anteriormente desenhados tiveram de ser inseridos na nova ferramenta. A notação da ferramenta utilizada é semelhante à que foi usada para definir os processos na sua forma inicial. De qualquer forma, o mapeamento não é directo. Uma vez inserida a informação dos processos, foi necessário realizar a parametrização de cada actividade.

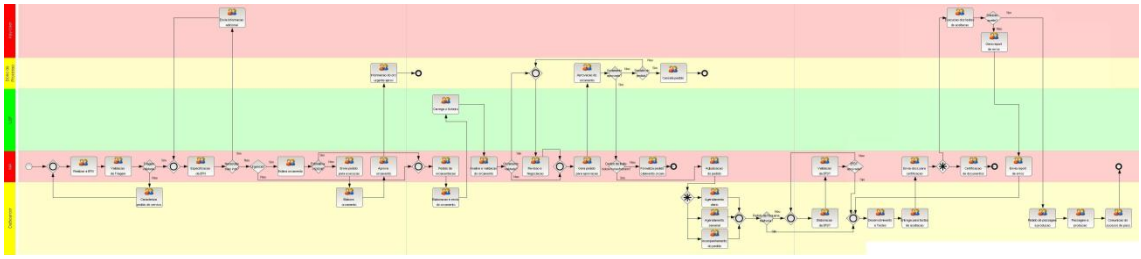
No final, foram desenhados em Savvion três processos: a Gestão de Alterações do ITIL (Fig. 27), o antigo e o novo processo da Manutenção Aplicacional (Fig. 28 e Fig. 29 respectivamente).



Fig, 27 - Processo de Gestão de Alterações



Fig, 28 – Processo de Manutenção Antigo



Fig, 29 – Processo de Manutenção Novo

6.2. Apresentação de resultados

Este capítulo apresenta os resultados pertencentes a cada um dos processos desenhados, num cenário com as seguintes características:

- Inserção de 10 instâncias no processo;
- Os intervalos de inserção, entre cada instância, são de 10 minutos;
- Todas as actividades definidas têm a duração média de 2 horas.

Processo de Gestão de Alterações – ITIL

Simulation Results for itil - (default)

Duration 282:00:00 Time

Process Time And Cost

Process	Scenario	Instance	Total Cost ($)	Waiting Time (Time)	Total Time (Time)
itil	(default)	10	NaN	1384:30:00	1802:30:00
		Grand Total	**NaN**	**1384:30:00**	**1802:30:00**

itil

Scenario (default)

Instances 10

Activity	Performer	Occurs	Waiting Time (Time)	Time to Complete (Time)	Total Time (Time)
Aprova ou Rejeita Alteracao1	Any member of Gestor de Alterações	4	**34:00:00**	8:00:00	42:00:00
Aprova ou Rejeita Alteracao2	Any member of Gestor de Alterações	5	**58:00:00**	10:00:00	68:00:00
Atribui prioridades iniciais	Any member of Gestor de Alterações	28	**276:00:00**	56:00:00	332:00:00
Circula RFCs entre CAB	Any member of Gestor de Alterações	4	**42:00:00**	8:00:00	50:00:00
Circula RFCs entre a Direccao	Any member of Gestor de Alterações	5	**54:00:00**	10:00:00	64:00:00
Convoca uma reuniao CAB e CAB CE	Any member of Gestor de Alterações	14	**154:00:00**	28:00:00	182:00:00
Coordena implementacao de alter	Any member of Gestor de Alterações	7	**56:00:00**	14:00:00	70:00:00
Coordena implementacao de altera	Any member of Gestor de Alterações	7	**66:00:00**	14:00:00	80:00:00
Cria alteracao e elabora planos	All member(s) of Criador de Alteracoes	14	**4:00:00**	28:00:00	32:00:00
Decide a categoria	Any member of Gestor de Alterações	14	**146:00:00**	28:00:00	174:00:00
Efectua Pedido	Utilizador	28	**82:30:00**	56:00:00	138:30:00
Fechado	Any member of Gestor de Alterações	1	**10:00:00**	2:00:00	12:00:00
Fechar	Any member of Gestor de Alterações	1	**8:00:00**	2:00:00	10:00:00
Filtra os pedidos	Any member of Gestor de Alterações	28	**232:00:00**	56:00:00	288:00:00
Garante actualizacao do registo	Any member of Gestor de Alterações	3	**20:00:00**	6:00:00	26:00:00
Informa CAB accao da alteracao	Any member of Gestor de Alterações	5	**52:00:00**	10:00:00	62:00:00
Plano de reposicao implementado	Any member of Gestor de Alterações	4	**36:00:00**	8:00:00	44:00:00
Planos de retorno implementados	All member(s) of Criador de Alteracoes	4	0:00:00	8:00:00	8:00:00
Prepara alteracoes com urgencia	All member(s) of Criador de Alteracoes	9	**2:00:00**	18:00:00	20:00:00
Reve alteracoes	Any member of Gestor de Alterações	3	**24:00:00**	6:00:00	30:00:00
Revisao de alteracoes	Any member of Gestor de Alterações	3	**24:00:00**	6:00:00	30:00:00
Testa alteracoes	All member(s) of Criador de Alteracoes	14	**2:00:00**	28:00:00	30:00:00
Teste urgente	All member(s) of Criador de Alteracoes	4	**2:00:00**	8:00:00	10:00:00

Resource	Unit	Cost/Unit	Threshold	Usage	Cost ($)
Any member of Gestor de Alterações	Hour	0	0	272:00:00	0
All member(s) of Criador de Alteracoes	Hour	0	0	90:00:00	0
Utilizador	Hour	0	0	56:00:00	0

Performers queue length and utilization

	Avg	Min	Max	Utilized(%)	Idle(%)
Any member of Gestor de Alterações	4,58	0	9	96,45	3,55
Any member of Gestor de Configurações	0	0	0	0	100
Value of 'Creator'	0	0	0	0	100
All member(s) of Criador de Alteracoes	0,04	0	1	31,91	68,09
Generic	0	0	0	0	100
Utilizador	0,29	0	9	19,86	80,14

Bottlenecks

Process	Activity	Performer	Avg Queue Length	Min Queue Length	Max Queue Length
itil	Aprova ou Rejeita Alteracao1	Any member of Gestor de Alterações	0,12	0	1
itil	Aprova ou Rejeita Alteracao2	Any member of Gestor de Alterações	0,21	0	1
itil	Atribui prioridades iniciais	Any member of Gestor de Alterações	0,98	0	5
itil	Circula RFCs entre CAB	Any member of Gestor de Alterações	0,15	0	1
itil	Circula RFCs entre a Direccao	Any member of Gestor de Alterações	0,19	0	1
itil	Convoca uma reuniao CAB e CAB CE	Any member of Gestor de Alterações	0,55	0	3
itil	Coordena implementacao de alter	Any member of Gestor de Alterações	0,2	0	1
itil	Coordena implementacao de altera	Any member of Gestor de Alterações	0,23	0	1
itil	Cria alteracao e elabora planos	All member(s) of Criador de Alteracoes	0,01	0	1
itil	Decide a categoria	Any member of Gestor de Alterações	0,52	0	3
itil	Efectua Pedido	Utilizador	0,29	0	9
itil	Fechado	Any member of Gestor de Alterações	0,04	0	1
itil	Fechar	Any member of Gestor de Alterações	0,03	0	1
itil	Filtra os pedidos	Any member of Gestor de Alterações	0,82	0	4
itil	Garante actualizacao do registo	Any member of Gestor de Alterações	0,07	0	1
itil	Informa CAB accao da alteracao	Any member of Gestor de Alterações	0,18	0	1
itil	Plano de reposicao implementado	Any member of Gestor de Alterações	0,13	0	1
itil	Prepara alteracoes com urgencia	All member(s) of Criador de Alteracoes	0,01	0	1
itil	Reve alteracoes	Any member of Gestor de Alterações	0,09	0	1
itil	Revisao de alteracoes	Any member of Gestor de Alterações	0,09	0	1
itil	Testa alteracoes	All member(s) of Criador de Alteracoes	0,01	0	1
itil	Teste urgente	All member(s) of Criador de Alteracoes	0,01	0	1

Tabela 3 - Processo de Gestão de Alterações - ITIL

Simulation Results for Gestao_da_Manutencao_Novo - (default)

Duration	232:00:00 Time

Process Time And Cost

Process	Scenario	Instance	Total Cost ($)	Waiting Time (Time)	Total Time (Time)
Gestao_da_Manutencao_Novo	(default)	10	NaN	1588:30:00	1830:30:00
		Grand Total	**NaN**	**1588:30:00**	**1830:30:00**

Gestao_da_Manutencao_Novo

Scenario	(default)
Instances	10

Activity	Performer	Occurs	Waiting Time (Time)	Time to Complete (Time)	Total Time (Time)
Acompanhamento do pedido	Outsourcer	2	**8:00:00**	4:00:00	12:00:00
Adjudicacao do pedido	MA	2	**16:00:00**	4:00:00	20:00:00
Agendamento diario	Outsourcer	2	0:00:00	4:00:00	4:00:00
Agendamento semanal	Outsourcer	2	**4:00:00**	4:00:00	8:00:00
Analise e validacao do orcamento	MA	7	**92:00:00**	14:00:00	106:00:00
Aprova orcamento	MA	3	**32:00:00**	6:00:00	38:00:00
Aprovacao do orcamento	Dono do Processo	10	0:00:00	20:00:00	20:00:00
Cancela pedido	Dono do Processo	2	**2:00:00**	4:00:00	6:00:00
Caracteriza pedido de servico	Outsourcer	10	0:00:00	20:00:00	20:00:00
Carrega o ficheiro	L&P	7	0:00:00	14:00:00	14:00:00
Certificacao de documentos	MA	3	**14:00:00**	4:00:00	18:00:00
Comunicao do sucesso de pass		0	0:00:00	0:00:00	0:00:00
Desenvolvimento e Testes	Outsourcer	5	**16:00:00**	10:00:00	26:00:00
Elabora orcamento	Outsourcer	3	0:00:00	6:00:00	6:00:00
Elaboracao da EFDT	Outsourcer	5	**16:00:00**	10:00:00	26:00:00
Elaboracao e envio do orcamento	Outsourcer	7	**6:00:00**	14:00:00	20:00:00
Entrega para testes de aceitacao	Outsourcer	5	**10:00:00**	10:00:00	20:00:00
Envia docs para certificacao	MA	5	**22:00:00**	6:00:00	28:00:00
Envia informacao adicional	Value of 'Creator'	9	0:00:00	18:00:00	18:00:00
Envia pedido para execucao	MA	3	**36:00:00**	6:00:00	42:00:00
Envia report de erros	MA	1	**8:00:00**	0:00:00	8:00:00
Especificacao da EFN	MA	19	**294:00:00**	38:00:00	332:00:00
Estima orcamento	MA	5	**76:00:00**	10:00:00	86:00:00
Execucao dos testes de aceitacao	Value of 'Creator'	3	0:00:00	6:00:00	6:00:00
Formaliza pedido cabimento orcam	MA	3	**20:00:00**	6:00:00	26:00:00
Gera pedido para aprovacao	MA	10	**120:00:00**	20:00:00	140:00:00
Gera report de erros	Value of 'Creator'	1	0:00:00	2:00:00	2:00:00
Informacao do orc urgente aprov	Dono do Processo	3	0:00:00	6:00:00	6:00:00
Passagem a producao		0	0:00:00	0:00:00	0:00:00
Pedido de orcamentacao	MA	7	**110:00:00**	14:00:00	124:00:00
Pedido de passagem a producao	Outsourcer	1	**2:00:00**	0:00:00	2:00:00
Realizar a EFN	MA	20	**238:30:00**	40:00:00	278:30:00
Revisao e Negociacao	MA	6	**68:00:00**	12:00:00	80:00:00
Validacao da EFDT	MA	5	**32:00:00**	8:00:00	40:00:00
Validacao da Triagem	MA	20	**346:00:00**	40:00:00	386:00:00

Resource	Unit	Cost/Unit	Threshold	Usage	Cost ($)
Outsourcer	Hour	0	0	82:00:00	0
Dono do Processo	Hour	0	0	30:00:00	0
Value of 'Creator'	Hour	0	0	26:00:00	0
L&P	Hour	0	0	14:00:00	0
MA	Hour	0	0	228:00:00	0

Performers queue length and utilization

	Avg	Min	Max	Utilized(%)	Idle(%)
Outsourcer	0,27	0	3	35,34	64,66
Dono do Processo	0,01	0	1	12,93	87,07
Value of 'Creator'	0	0	0	11,21	88,79
L&P	0	0	0	6,03	93,97
Generic	0	0	0	0	100
CT	0	0	0	0	100
MA	6,57	0	9	98,28	1,72

Bottlenecks

Process	Activity	Performer	Avg Queue Length	Min Queue Length	Max Queue Length
Gestao_da_Manutencao_Novo	Acompanhamento do pedido	Outsourcer	0,03	0	1
Gestao_da_Manutencao_Novo	Adjudicacao do pedido	MA	0,07	0	1
Gestao_da_Manutencao_Novo	Agendamento semanal	Outsourcer	0,02	0	1
Gestao_da_Manutencao_Novo	Analise e validacao do orcamento	MA	0,4	0	2
Gestao_da_Manutencao_Novo	Aprova orcamento	MA	0,14	0	1
Gestao_da_Manutencao_Novo	Cancela pedido	Dono do Processo	0,01	0	1
Gestao_da_Manutencao_Novo	Certificacao de documentos	MA	0,06	0	1
Gestao_da_Manutencao_Novo	Desenvolvimento e Testes	Outsourcer	0,07	0	1
Gestao_da_Manutencao_Novo	Elaboracao da EFDT	Outsourcer	0,07	0	1
Gestao_da_Manutencao_Novo	Elaboracao e envio do orcamento	Outsourcer	0,03	0	1
Gestao_da_Manutencao_Novo	Entrega para testes de aceitacao	Outsourcer	0,04	0	1
Gestao_da_Manutencao_Novo	Envia docs para certificacao	MA	0,09	0	1
Gestao_da_Manutencao_Novo	Envia pedido para execucao	MA	0,16	0	1
Gestao_da_Manutencao_Novo	Envia report de erros	MA	0,03	0	1
Gestao_da_Manutencao_Novo	Especificacao da EFN	MA	1,27	0	5
Gestao_da_Manutencao_Novo	Estima orcamento	MA	0,33	0	1
Gestao_da_Manutencao_Novo	Formaliza pedido cabimento orcam	MA	0,09	0	1
Gestao_da_Manutencao_Novo	Gera pedido para aprovacao	MA	0,52	0	3
Gestao_da_Manutencao_Novo	Pedido de orcamentacao	MA	0,47	0	2
Gestao_da_Manutencao_Novo	Pedido de passagem a producao	Outsourcer	0,01	0	1
Gestao_da_Manutencao_Novo	Realizar a EFN	MA	1,03	0	9
Gestao_da_Manutencao_Novo	Revisao e Negociacao	MA	0,29	0	2
Gestao_da_Manutencao_Novo	Validacao da EFDT	MA	0,14	0	2
Gestao_da_Manutencao_Novo	Validacao da Triagem	MA	1,49	0	9

Tabela 4 - Processo de Manutenção Aplicacional - Novo

Processo de Manutenção Aplicacional – Antigo

Simulation Results for Manutencao_Antigo - (default)

Duration 252:00:00 Time

Process Time And Cost

Process	Scenario	Instance	Total Cost ($)	Waiting Time (Time)	Total Time (Time)
Manutencao_Antigo	(default)	10	NaN	1598:30:00	1870:30:00
		Grand Total	**NaN**	**1598:30:00**	**1870:30:00**

Manutencao_Antigo

Scenario (default)

Instances 10

Activity	Performer	Occurs	Waiting Time (Time)	Time to Complete (Time)	Total Time (Time)
Adjudicacao do pedido	Any member of MA	5	**46:00:00**	10:00:00	56:00:00
Agendamento diario	Any member of Outsourcer	5	**6:00:00**	10:00:00	16:00:00
Agendamento semanal	Any member of Outsourcer	3	0:00:00	6:00:00	6:00:00
Analise e validacao do orcamento	Any member of MA	7	**70:00:00**	14:00:00	84:00:00
Aprova orcamento	Any member of MA	3	**22:00:00**	6:00:00	28:00:00
Aprova pedido	Dono do Processo	10	0:00:00	20:00:00	20:00:00
Aprovacao do orcamento	Dono do Processo	10	0:00:00	20:00:00	20:00:00
Cancela pedido	Dono do Processo	2	0:00:00	4:00:00	4:00:00
Caracteriza o tipo de servico	Any member of L&P	25	**178:00:00**	50:00:00	228:00:00
Carrega ficheiro no ITG	Any member of L&P	7	**16:00:00**	14:00:00	30:00:00
Certificacao da documentacao	Any member of MA	5	**40:00:00**	10:00:00	50:00:00
Comunicacao do sucesso de passag	Any member of Outsourcer	2	0:00:00	4:00:00	4:00:00
Desenvolvimento e Testes	Any member of Outsourcer	5	**4:00:00**	10:00:00	14:00:00
Elabora e envia orcamento	Any member of Outsourcer	3	0:00:00	6:00:00	6:00:00
Elaboracao da EFDT	Any member of Outsourcer	5	**10:00:00**	10:00:00	20:00:00
Elaboracao e envio de orcamento	Any member of Outsourcer	7	0:00:00	14:00:00	14:00:00
Entrega para testes	Any member of Outsourcer	5	**6:00:00**	10:00:00	16:00:00
Envia e carrega docs para certif	Any member of MA	5	**32:00:00**	10:00:00	42:00:00
Envia informacao adicional	Any member of (Value of 'Creator')	10	0:00:00	20:00:00	20:00:00
Envia pedido para execucao	Any member of MA	3	**28:00:00**	6:00:00	34:00:00
Envia report de erros	Any member of MA	3	**20:00:00**	0:00:00	20:00:00
Especificacao da EFN	Any member of MA	24	**250:00:00**	42:00:00	292:00:00
Estima orcamento	Any member of MA	5	**64:00:00**	10:00:00	74:00:00
Execucao dos testes de aceitacao	Any member of (Value of 'Creator')	5	0:00:00	10:00:00	10:00:00
Gera pedido para aprovacao	Any member of MA	10	**108:00:00**	20:00:00	128:00:00
Gera report de erros	Any member of (Value of 'Creator')	3	0:00:00	6:00:00	6:00:00
Identifica o tipo de servico	Any member of L&P	10	**72:00:00**	20:00:00	92:00:00
Informacao do orc urgente aprov	Dono do Processo	3	0:00:00	6:00:00	6:00:00
Passagem a producao	Any member of Outsourcer	2	**2:00:00**	4:00:00	6:00:00
Pedido de orcamentacao	Any member of MA	8	**92:00:00**	14:00:00	106:00:00
Pedido de passagem a producao	Any member of Outsourcer	2	**2:00:00**	4:00:00	6:00:00
Programacao do servico	Any member of L&P	20	**90:00:00**	38:00:00	128:00:00
Recebe agendamento e prioridades	Any member of L&P	8	**6:00:00**	16:00:00	22:00:00
Regista pedido no ITG	Any member of (Value of 'Creator')	10	**82:30:00**	20:00:00	102:30:00
Revisao e Negociacao	Any member of MA	7	**68:00:00**	14:00:00	82:00:00
Validacao da EFDT	Any member of MA	5	**40:00:00**	4:00:00	44:00:00
Validacao da triagem	Any member of MA	30	**244:00:00**	58:00:00	302:00:00

Resource	Unit	Cost/Unit	Threshold	Usage	Cost ($)
Dono do Processo	Hour	0	0	50:00:00	0
Any member of L&P	Hour	0	0	138:00:00	0
Any member of MA	Hour	0	0	218:00:00	0
Any member of (Value of 'Creator')	Hour	0	0	56:00:00	0
Any member of Outsourcer	Hour	0	0	78:00:00	0

Performers queue length and utilization

	Avg	Min	Max	Utilized(%)	Idle(%)
Generic	0	0	0	0	100
Dono do Processo	0	0	0	19,84	80,16
Any member of L&P	1,44	0	9	54,76	45,24
Any member of MA	4,46	0	11	86,51	13,49
Any member of (Value of 'Creator')	0,33	0	9	22,22	77,78
Any member of Outsourcer	0,12	0	2	30,95	69,05

Bottlenecks

Process	Activity	Performer	Avg Queue Length	Min Queue Length	Max Queue Length
Manutencao_Antigo	Adjudicacao do pedido	Any member of MA	0,18	0	1
Manutencao_Antigo	Agendamento diario	Any member of Outsourcer	0,02	0	1
Manutencao_Antigo	Analise e validacao do orcamento	Any member of MA	0,28	0	2
Manutencao_Antigo	Aprova orcamento	Any member of MA	0,09	0	1
Manutencao_Antigo	Caracteriza o tipo de servico	Any member of L&P	0,71	0	5
Manutencao_Antigo	Carrega ficheiro no ITG	Any member of L&P	0,06	0	1
Manutencao_Antigo	Certificacao da documentacao	Any member of MA	0,16	0	2
Manutencao_Antigo	Desenvolvimento e Testes	Any member of Outsourcer	0,02	0	1
Manutencao_Antigo	Elaboracao da EFDT	Any member of Outsourcer	0,04	0	2
Manutencao_Antigo	Entrega para testes	Any member of Outsourcer	0,02	0	1
Manutencao_Antigo	Envia e carrega docs para certif	Any member of MA	0,13	0	2
Manutencao_Antigo	Envia pedido para execucao	Any member of MA	0,11	0	1
Manutencao_Antigo	Envia report de erros	Any member of MA	0,08	0	1
Manutencao_Antigo	Especificacao da EFN	Any member of MA	0,99	0	3
Manutencao_Antigo	Estima orcamento	Any member of MA	0,25	0	1
Manutencao_Antigo	Gera pedido para aprovacao	Any member of MA	0,43	0	2
Manutencao_Antigo	Identifica o tipo de servico	Any member of L&P	0,29	0	4
Manutencao_Antigo	Passagem a producao	Any member of Outsourcer	0,01	0	1
Manutencao_Antigo	Pedido de orcamentacao	Any member of MA	0,37	0	2
Manutencao_Antigo	Pedido de passagem a producao	Any member of Outsourcer	0,01	0	1
Manutencao_Antigo	Programacao do servico	Any member of L&P	0,36	0	3
Manutencao_Antigo	Recebe agendamento e prioridades	Any member of L&P	0,02	0	1
Manutencao_Antigo	Regista pedido no ITG	Any member of (Value of 'Creator')	0,33	0	9
Manutencao_Antigo	Revisao e Negociacao	Any member of MA	0,27	0	2
Manutencao_Antigo	Validacao da EFDT	Any member of MA	0,16	0	1
Manutencao_Antigo	Validacao da triagem	Any member of MA	0,97	0	4

Tabela 5 - Processo de Manutenção Aplicacional - Antigo

6.3. Análise de resultados

A tabela seguinte compara os processos que foram desenhados através de quatro parâmetros relevantes para a análise:

	Gestão de Alterações	MA – Novo	MA - Antigo
Número de Actividades	23	35	37
Bottlenecks	25	29	30
Duração	12 dias	9 dias e 16 horas	10 dias e 12 horas
Total de Horas (10 pedidos)	1802:30:00	1830:30:00	1870:30:00

Tabela 6 – Tabela comparativa de processos

A partir da tabela 5 podemos analisar o seguinte:

- A passagem da nomenclatura QPR para a aplicação Savvion provocou o aumento do número de actividade do processo MA-Novo – inicialmente apresentava 26 actividades. Em comparação, o processo MA-Antigo não sofreu aumentos na nova nomenclatura, visto que possuía 34 actividades;
- Os processos MA-Novo e MA-Antigo apresentam, respectivamente, 29 e 30 *bottlenecks.* Como seria de esperar, o processo que teve melhor desempenho foi o MA-Novo. O tempo total do processo foi melhorado em cerca de 30 minutos, no caso de 10 instâncias;
- O DSI tem cerca de 3000 pedidos mensais, o que significa uma redução de 144 horas por mês (cerca de 6 dias) com o MA-Novo;
- Se executássemos os pedidos individualmente, precisaríamos de 76 dias para o processo MA-Novo e 78 dias para o processo MA-Antigo;
- No processo de Gestão de Alterações:
  - Se os pedidos fossem executados individualmente, demorariam menos tempo que ambos os processos MA;
  - A duração é superior em ambos, devido ao tempo de espera considerado em actividades mortas (por exemplo “informa Conselho Consultivo de Alterações” – CAB);

7. Conclusão

Considerando o estado da arte da manutenção de software e o caso de estudo realizado, é possível relacionar a teoria com a prática e avaliar o sucesso da proposta.

A proposta para a optimização do processo de manutenção de software consistiu no desenho de um processo baseado em boas práticas, com a particularidade e vantagem de ser implementado em *outsourcing*.

Após a análise comparativa entre os processos de manutenção, antigo e novo, podemos concluir que o facto de os processos estarem em *outsourcing* não afecta o desempenho dos mesmos.

O ITIL pode ser aplicado em empresas que operam em *full-outsourcing* e, visto ser o factor variante entre o processo antigo e o novo, tem um impacto notório sobre o processo afectado. O impacto refere-se ao número de horas que irá ser poupado, tanto a curto, como a médio e longo prazo.

É necessário chamar a atenção das empresas para as ineficiências nos processos que resultam da rotatividade de pessoas nesta área. É preocupante o esforço desempenhado nesta fase caso não seja produzida documentação com qualidade.

As empresas têm o dever de incutir a qualidade tanto nos documentos produzidos, como na forma como desenvolvem a sua actividade. Devem começar por enraizar as boas práticas nos seus processos, podendo beneficiar com esta opção, como foi demonstrado neste estudo.

7.1. Trabalho Futuro

A área de Manutenção de Software tem vindo a ganhar ênfase. Durante o tempo de realização deste estudo, realizei diversas pesquisas e o número de estudos sobre este tema tem vindo aumentar.

Com o terceiro capítulo deste estudo tentei enunciar a maioria dos problemas existentes nesta área. Chegando ao fim deste estudo, encontrei alguns documentos sobre a manutenção de software que se referem ao tema, como uma solução.

A manutenção, como solução para problemas encontrados na fase pós-desenvolvimento, é certamente um tema a explorar.

Com este estudo, conclui-se que o tema da manutenção de software pode, e deve, ser alvo de mais estudos.

Referências

20000-1 ISO/IEC ISO/IEC 20000-1, Information Technology. - 2005.

April A. [et al.] Software Maintenance in a Service Level Agreement : Controlling the Customer Expectations [Conferência] // Fourth European Software Conference, FESMA. - Heidleberg, Germany : [s.n.], 2001. - pp. 39-47.

April Alain [et al.] Software Maintenance Maturity Model (SMmm): the software maintenance process model [Jornal] // Journal of Software Maintenance and Evolution: Research and Practice. - May de 2005. - 3 : Vol. 17. - pp. 197-223.

Bandinelli S. C., Fuggetta A. e Ghezzi C. Software Process Model Evolution in the SPADE Environment [Jornal]. - [s.l.] : IEEE Transactions on Software Engineering, 1993. - 12 : Vol. 19. - pp. 1128-1144.

Banker R.D. [et al.] Software Complexity and Maintenance Costs [Jornal]. - [s.l.] : Comm. ACM, Nov de 1993. - Vol. 36. - pp. 81-94.

Bengtsson P.O. e Bosch J. Architecture Level Prediction of Software Maintenance [Conferência] // Third European Conf. Software Maintenance and Reeng. - 1999. - pp. 139-147.

Bennet P. Lientz e Swanson E. Burton Problems in application software maintenance [Jornal] // Communications of the ACM. - [s.l.] : ACM, 1981. - 11 : Vol. 24. - pp. 763-769.

Bennett Keith H e Rajlich Václav T Software maintenance and evolution: a roadmap [Conferência] // Conference on The Future of Software Engineering. - Limerick, Ireland : [s.n.], 2000. - pp. 73-87.

Boehm B.W. A Spiral Model of Software Development and Enhancement [Conferência] // Computer. - 1988. - pp. 61-72.

Boehm B.W. Software Risk Management: Principles and Practices [Jornal]. - [s.l.] : IEEE Software, 1991. - 1 : Vol. 8. - pp. 32-41.

Car Z. e Mikac B. A method for modeling and evaluating software maintenance process performances [Conferência] // Sixth European Conference on Software Maintenance and Reengineering. - 2002. - pp. 15-23.

Carr Mahil e Wagner Christian A Study of Reasoning Processes in Software Maintenance Management [Jornal] // Information Technology Management. - 2002. - pp. 181-203.

Chapin Ned [et al.] Types of software evolution and software maintenance [Jornal] // Journal of Software Maintenance and Evolution: Research and Practice. - [s.l.] : John Wiley & Sons, Ltd, 2001. - 1 : Vol. 13. - pp. 3-30.

Davis Alan M. e Bersoff Edward H. Impacts of life cycle models on software configuration management [Artigo] // Communications of the ACM. - [s.l.] : ACM, 1991. - 8 : Vol. 34. - pp. 104-118.

Foster JR Cost Factors in Software Maintenance [Livro]. - [s.l.] : University of Durham, 1993.

Fuggetta Alfonso Software process: a roadmap [Conferência] // Conference on The Future of Software Engineering. - Limerick, Ireland : [s.n.], 2000. - pp. 25-34.

Gartner Gartner Symposium ITxpo 2000 [Conferência]. - Cannes, France : [s.n.], 2000.

Gartner Gartner Symposium ITxpo 2002 [Conferência]. - Tokyo, Japan : [s.n.], 2002.

Glass Rorbert L e Hunt Andrew Software Conflict 2.0: The Art And Science of Software Engineering [Secção do Livro]. - [s.l.] : Developer.* Books, 2006.

IEEE IEEE STD 1219: Standard for Software Maintenance [Jornal]. - 1998.

ITSMF-NL Foundations of IT Service Management [Livro]. - [s.l.] : Van Haren Publishing, 2005.

Lientz B. P., Swanson E. B. e Tompkins G. E. Characteristics of application software maintenance [Jornal] // Communications of the ACM. - 1978. - 6 : Vol. 21. - pp. 466-471.

Lientz Bennett P. e Swanson E. Burton Software Maintenance Management [Jornal]. - Boston, MA : Addison-Wesley Longman Publishing , 1980.

Mamone S. The IEEE Standard for Software Maintenance [Jornal] // Software. - 1994. - 1 : Vol. 19. - pp. 75-76.

Martin RJ e Osborne WM Guidance On Software Maintenance [Livro]. - [s.l.] : Government Printing Office, 1983.

Office of Government Commerce Applications Management: Itil (It Infrastructure Library Series) [Livro]. - [s.l.] : Stationery Office, 2002.

OutSystems [Online]. - OutSystems. - 15 de Setembro de 2007. - http://www.outsystems.pt/.

Pankaj Bhatt [et al.] “Influencing factors in outsourced software maintenance [Jornal] // ACM SIGSOFT Software Engineering Notes. - 2006. - 3 : Vol. 31. - pp. 1-6.

Pigoski Tom [et al.] “How Much Has Software Maintenance Changed Since 1983? [Jornal] // 12th International Conference on Software Maintenance (ICSM'96), . - 1996. - p. 34.

QPR Software Plc [Online]. - QPR Software Plc, 1991. - 15 de Setembro de 2007. - http://www.qpr.com/.

Savvion, Inc [Online]. - http://www.savvion.com/.

Savvion, Inc [Online]. - 15 de Setembro de 1999. - 2007. - http://www.savvion.com.

Swanson E. Burton The dimensions of maintenance [Conferência] // 2nd International Conference on Software Engineering. - San Francisco, California, United States : [s.n.], 1976. - pp. 492-497.

Anexos

Anexo 1 – Documento de Especificação Funcional

1. Gestor de Alterações

O Gestor de Alterações tem como ecrã inicial a Listagem do RFCs disponíveis no sistema. Através deste ecrã inicial é possível chegar a todas as restantes funcionalidades. O caminho percorrido para chegar às funcionalidades finais está assinalado na zona superior do ecrã.

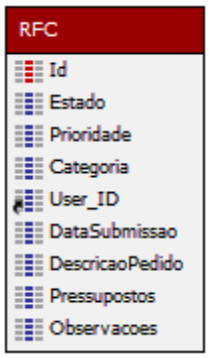
Listar RFCs

Esta funcionalidade é apresentada no ecrã inicial do sistema. Lista todos os RFCs criados, indepentemente do seu estado.

- **Requisitos de dados**

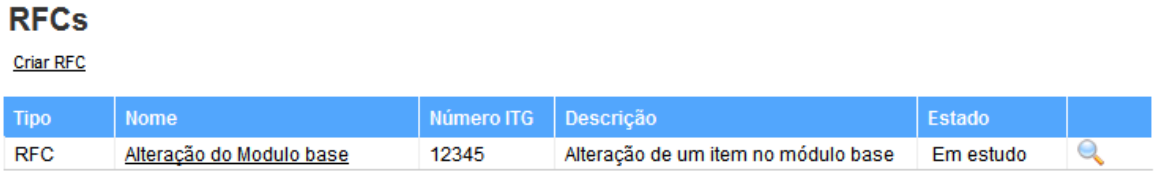
O diagrama seguinte é parte do Diagrama Lógico da BD, presente no Documento de Especificação Técnica da GA (Secção 4.1.2).

A entidade "RFC" e atributos, necessários à execução desta funcionalidade está representada na figura seguinte:



- **Interface com o Utilizador**

A listagem dos RFCs é apresentada em forma de tabela. Esta tabela é composta pelo **Tipo** (Meta-CI RFC), pelo **Nome** e **Descrição** associadas ao RFC e também o **Estado** actual em que o RFC se encontra.



Através deste ecrã é possível aceder a duas funcionalidades:

- Criação do RFC, premindo o link 'Criar RFC' que se encontra no canto superior esquerdo do ecrã;
- Visualização dos detalhes do RFC, premindo o ícone Lupa (), apresentado na última coluna do lado direito, ou premindo o nome do RFC que se encontra sob a forma de link.

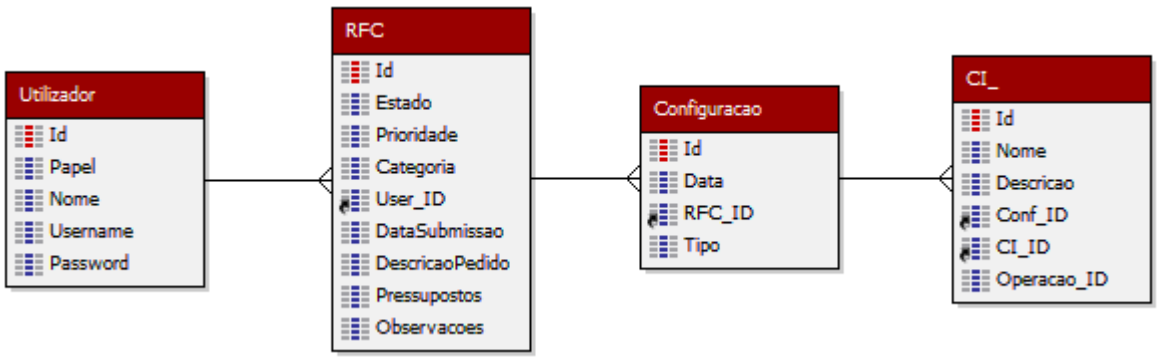
CRIAR RFC

Esta funcionalidade permite ao utilizador criar o RFC tendo como base, dados existentes na CMDB.

- Requisitos de dados

O diagrama seguinte é parte do Diagrama Lógico da BD presente no Documento de Especificação Técnica da GA (Secção 4.1.2).

A entidade "RFC" e atributos, necessários à execução desta funcionalidade está representada na figura seguinte:



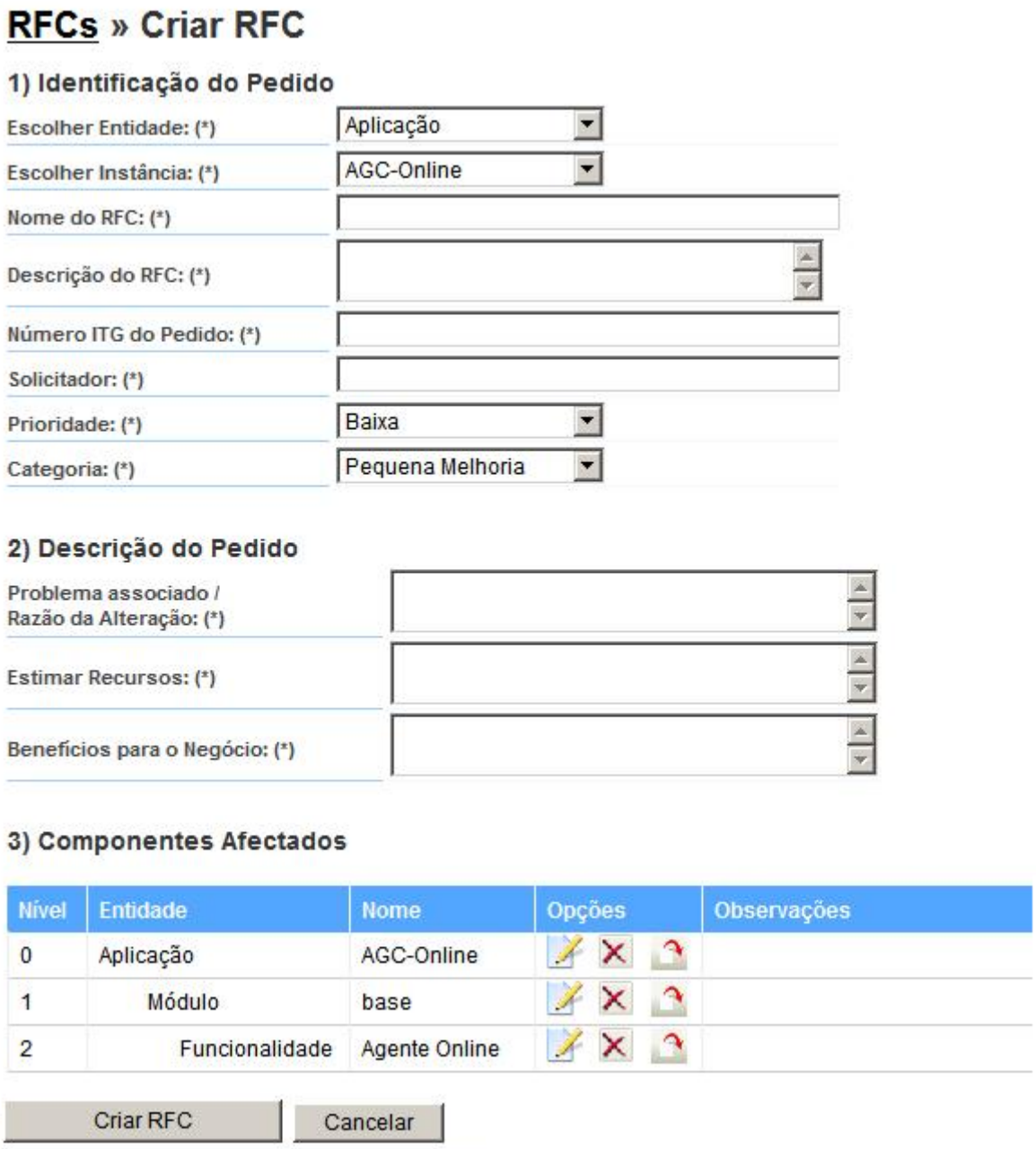
- Interface com o Utilizador

A criação de um RFC passa por preencher as seguintes secções:

1) Identificação do Pedido, secção onde são inseridos dados vindos do ITG (número ITG, nome e descrição do Pedido, solicitador, prioridade e categoria) e dados vindos da CMDB (Entidade e Instância);

2) Descrição do Pedido, secção que corresponde à descrição do pedido propriamente dita, como a Razão da Alteração, Benefícios para o Negócio e Estimação de Recursos.

3) Pressupostos do Pedido, esta secção corresponde a uma visualização gráfica com maior nível de detalhe da Instância (CI) especificada na secção 1). Nesta secção, é apresentada a Instância, e caso existam, as suas relações, e sub-relações, de inclusão.



- Nível – grau de inclusão. Lê-se da seguinte forma: A aplicação é composta por módulos; o módulo é composto por funcionalidades;
- Entidade – nome do Meta-CI;
- Nome – nome do CI;

- Opções de Alteração do CI:
  -  - permite alterar o CI em questão;
  -  - permite apagar o CI em questão;
  -  - permite criar uma relação de inclusão com um novo CI do nível imediatamente inferior;
- Observações - o utilizador pode optar por descrever a necessidade, ou especificar detalhadamente a opção que aplica ao CI em questão.

A criação do RFC, vai ser reflectida na CMDB, estando logo disponível na Listagem dos RFCs.

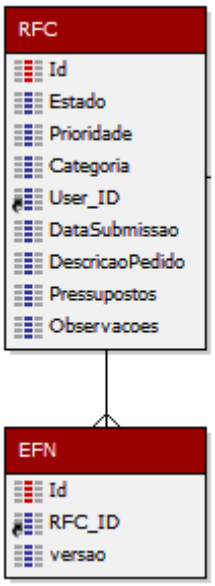
DETALHES RFC

Esta funcionalidade permite ver os detalhes associados ao RFC, escolhido pelo utilizador. Esta funcionalidade pode ser acedida através do ecrã que contem a listagem dos RFCs, ou seja do ecrã principal.

- Requisitos de dados

O diagrama seguinte é parte do Diagrama Lógico da BD presente no Documento de Especificação Técnica da GA (Secção 4.1.2).

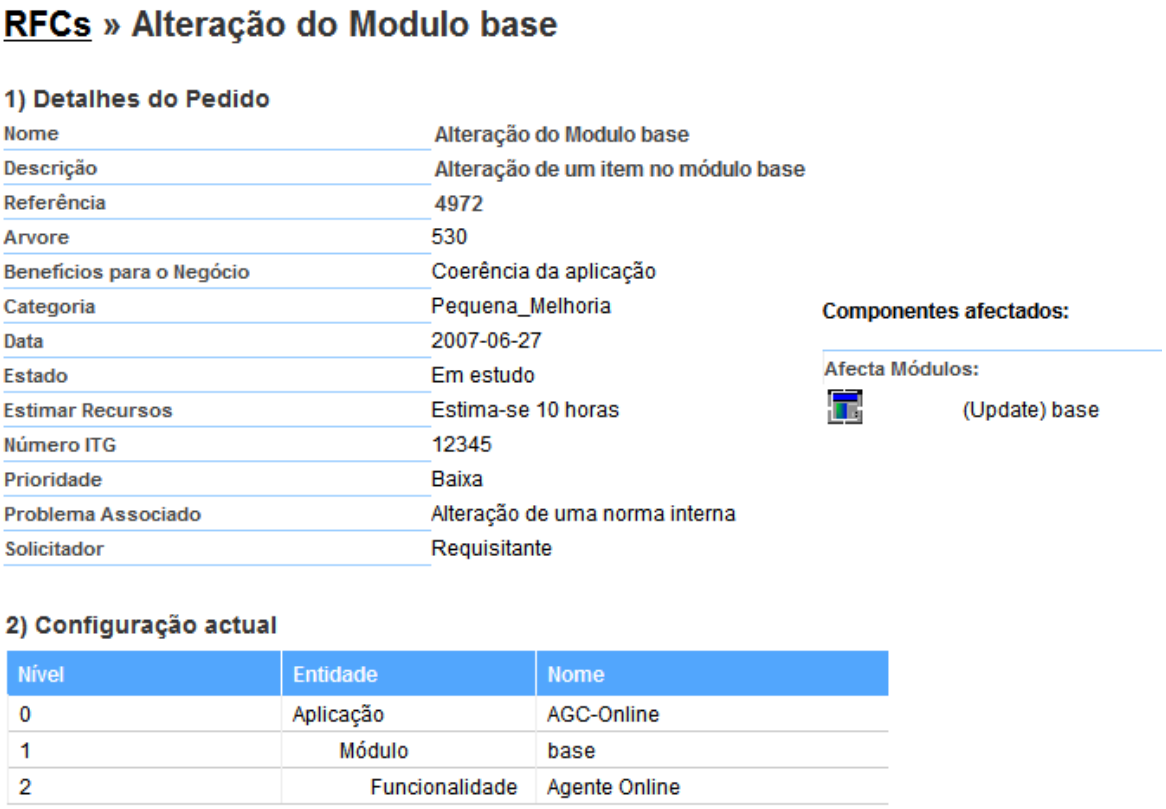
A entidade "RFC" e atributos, necessários à execução desta funcionalidade está representada na figura seguinte:



- **Interface com o Utilizador**

Esta funcionalidade permite ao utilizador:

- Visualizar todos os detalhes do RFC enunciados na criação do mesmo, incluindo os pressupostos de alteração (lado direito do ecrã);
- Visualizar os campos não inicializados pelo utilizador, como: a Data em que o RFC foi criado, o Estado actual do RFC, etc.



G do ecrã.

Para voltar ao menu anterior, o utilizador deve premir '**RFCs**' no canto superior esquerdo.

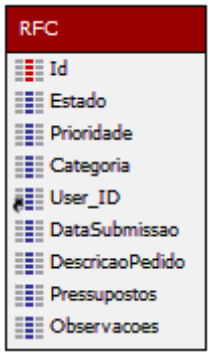
EDDITAR RFC

Esta funcionalidade permite ao utilizador editar as propriedades do RFC, definidas quando o mesmo RFC foi criado. A nova informação é guardada e fica logo disponível para consulta. Esta funcionalidade está disponível a partir do ecrã 'Detalhes RFC' (4.2.3). O campo Estado é o campo de maior relevo da edição do RFC, visto que vai desencadear outras funcionalidades, no futuro, tais como adicionar novos documentos ao RFC (EFN e Orçamento).

- **Requisitos de dados**

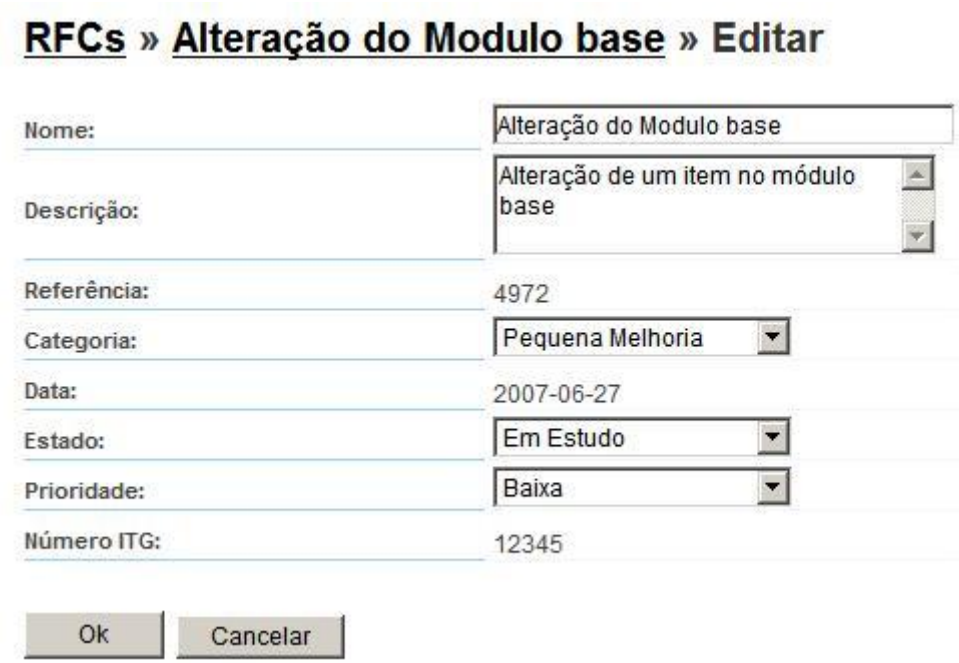
O diagrama seguinte é parte do Diagrama Lógico da BD presente no Documento de Especificação Técnica da GA (Secção 4.1.2).

A entidade "RFC", necessária à execução desta funcionalidade está representada na figura seguinte:



- **Interface com o Utilizador**

Esta funcionalidade permite visualizar todos os campos associados ao RFC. Os atributos que não podem ser alterados, tais como a Data e o NumITG, encontram-se bloqueados ao utilizador. Os restantes campos podem ser alterados ao critério do utilizador.



Pa ferior esque otão "Can

O s' no canto

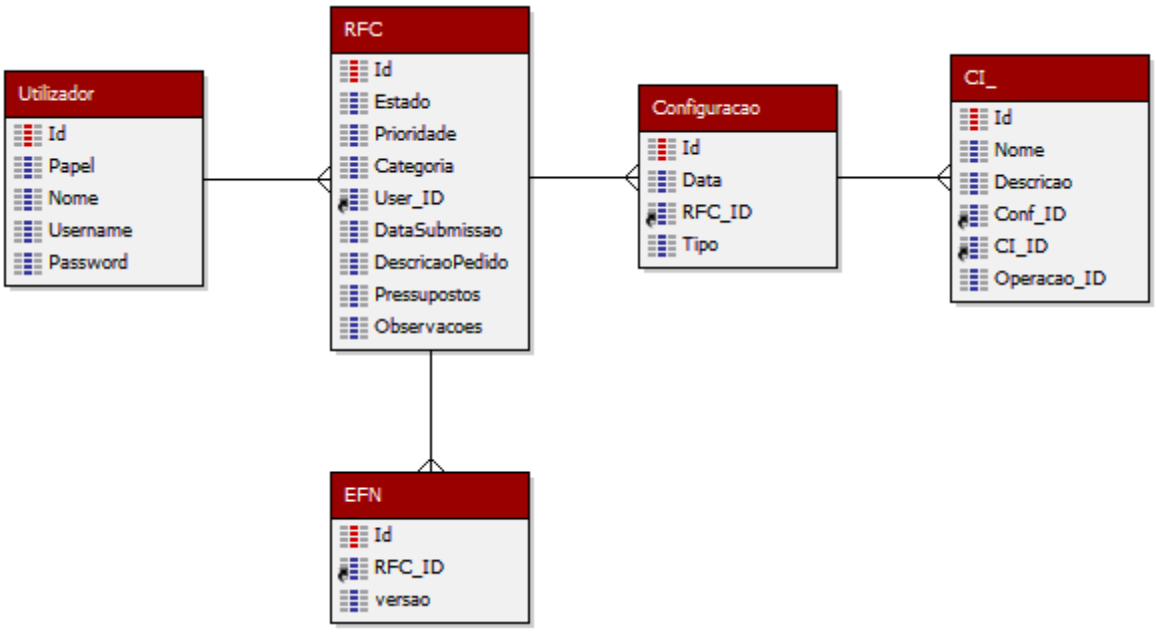
GERA

Pa esta funcionalidade está disponível através do ecrã "Detalhes RFC" associado a um RFC. A opção de Gerar um RFC permite gerar um documento que inclui toda a informação disponível, associada ao RFC em questão. O documento gerado pode ser guardado ou apenas visualizado, pelo utilizador.

- **Requisito de Dados**

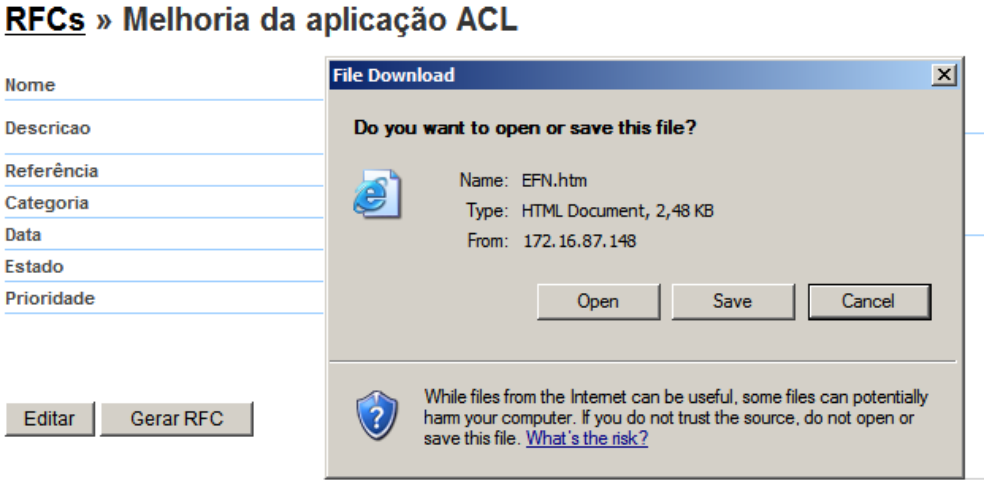
O diagrama seguinte é parte do Diagrama Lógico da BD presente no Documento de Especificação Técnica da GA (Secção 4.1.2).

A entidade "RFC" e atributos, necessários à execução desta funcionalidade está representada na figura seguinte:

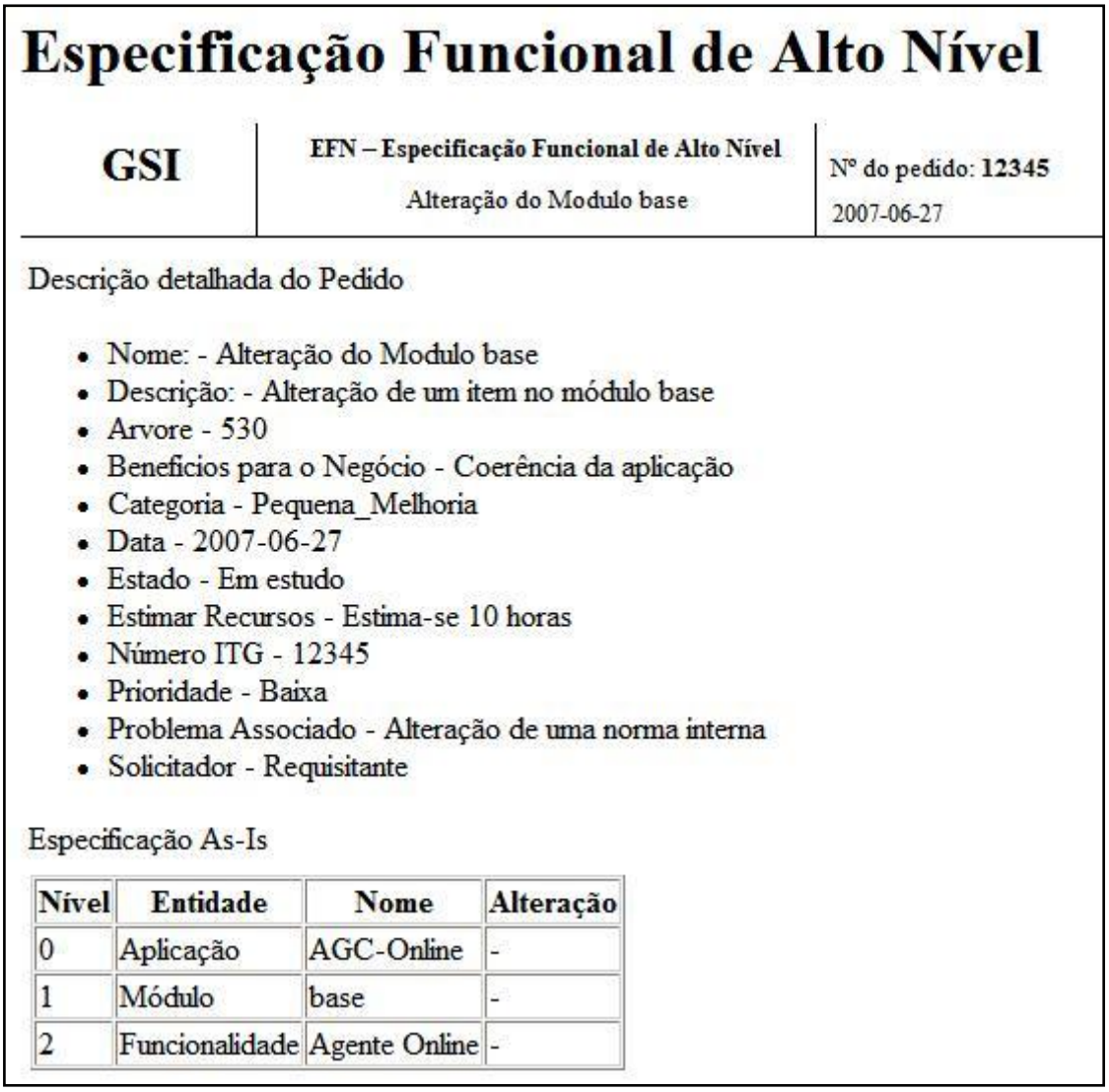


- **Interface com o Utilizador**

Ao premir o botão 'Gerar RFC', no canto inferior esquerdo, é disponibilizado um ecrã do Windows onde o utilizador pode escolher 'Open' ou 'Save', conforme desejar Abrir ou Salvar o documento. Caso o utilizador pretenda cancelar a geração do RFC deverá premir a opção 'Cancel'.



O documento gerado contém a informação associada ao RFC. O documento tem a seguinte apresentação:



Especificação Funcional de Alto Nível

GSI	EFN – Especificação Funcional de Alto Nível Alteração do Modulo base	Nº do pedido: 12345 2007-06-27

Descrição detalhada do Pedido

- Nome: - Alteração do Modulo base
- Descrição: - Alteração de um item no módulo base
- Arvore - 530
- Beneficios para o Negócio - Coerência da aplicação
- Categoria - Pequena_Melhoria
- Data - 2007-06-27
- Estado - Em estudo
- Estimar Recursos - Estima-se 10 horas
- Número ITG - 12345
- Prioridade - Baixa
- Problema Associado - Alteração de uma norma interna
- Solicitador - Requisitante

Especificação As-Is

Nível	Entidade	Nome	Alteração
0	Aplicação	AGC-Online	-
1	Módulo	base	-
2	Funcionalidade	Agente Online	-

O utilizador também pode aceder ao ecrã inicial. Para tal, deve premir '**RFCs**' no canto superior esquerdo do ecrã.

Anexo 2 – Documento de Especificação Técnica

1. Especificação de Interface

INTERFACES EXTERNOS

GESTOR DE ALTERAÇÕES

Os interfaces externos – é o acesso de uma aplicação externa ao núcleo CMDB, ou à camada aplicacional, funcionando também no sentido oposto.

Os interfaces criados estão definidos na Especificação de Requisitos Funcionais. Esses interfaces são o acesso pelo qual utilizadores, ou outras aplicações, acedem ao módulo GA. Para uma melhor integração entre duas aplicações, deveria existir uma uniformização das mensagens trocadas entre os sistemas, caso se trate de duas tecnologias diferentes.

As acções subjacentes aos interfaces implicam a integração entre os módulos. Isto pode ser conseguido através da uniformização das mensagens trocadas entre os sistemas.

Os interfaces que existem são:

- Listar RFCs
- Criar RFC
- Detalhes RFC
- Editar RFC

Os interfaces externos que poderiam existir, são:

- Permitir a visualização de informação relativa de um pedido, disponível em ITG, através do número ITG desse mesmo pedido. Esta funcionalidade poderia estar disponível nas interfaces em que é disponibilizado o número ITG de um pedido (Detalhes, Editar, Listar).

GESTOR DE TIPIFICADOS

Os interfaces externos são o acesso de uma aplicação externa ao núcleo CMDB, ou à camada aplicacional, funcionando também no sentido oposto.

Os interfaces criados estão definidos na Especificação de Requisitos Funcionais, secção X.XX. Esses interfaces são o acesso pelo qual utilizadores, ou outras aplicações, acedem ao módulo ProcITG. Para uma melhor integração entre duas aplicações, deveria existir uma uniformização das mensagens trocadas entre os sistemas, caso se trate de duas tecnologias diferentes.

As acções subjacentes aos interfaces implicam a integração entre os módulos. Isto pode ser conseguido através da uniformização das mensagens trocadas entre os sistemas.

Os interfaces que existem são:

- Criar Tipo de Pedido Tipificado (Backoffice)
- Criar Pedido Tipificado
- Detalhes Pedido Tipificado
- Pesquisar Pedidos Tipificados
- Editar Pedido Tipificado

Os interfaces externos que poderiam existir, são:

- Permitir a visualização de informação relativa de um pedido, disponível em ITG, através do número ITG desse mesmo pedido. Esta funcionalidade poderia estar disponível nas interfaces em que é disponibilizado o numero ITG de um pedido (Detalhes, Editar, Listar)

INTERFACES INTERNOS

GESTOR DE ALTERAÇÕES

A comunicação entre a GA e a CMDB, implica a transferência de dados. Posto isto, quando as funcionalidades das CMDB são utilizadas, os dados de retorno tem de ser lidos, convertidos e interpretados para poderem ser entendidos pela GA. O método utilizado para a integração dos dois módulos é: fazer uma estrutura de dados idêntica à estrutura da CMDB, permitindo assim que os dados obtidos, a através de invocação a chamadas externas, possam ter um mapeando mecânico.

Os dados que passam entre as duas aplicações são os outputs das funções externas invocadas pela GA durante a sua execução. Estão listadas as funções, cujos outputs são utilizados para a execução da GA, e a sua correspondência na GA. Os detalhes estão no dicionário de dados. Essas funções são as seguintes:

Quadro: Função CMDB out = dados GA

- C_Relacao – sem correspondência
- L_CIs -> devolve uma lista de 'Instancia'
- L_MetaCIs → devolve uma lista de 'CI'
- R_CI → devolve uma estrutura do tipo 'instanciaDetails_t'
- U_Value – sem correspondência
- UC_CI - sem correspondência

GESTOR DE TIPIFICADOS

A comunicação entre a ProcITG e a CMDB, implica transferência de dados. Posto isto, quando as funcionalidades das CMDB são utilizadas, os dados de retorno tem de ser lidos, convertidos e interpretados para poderem ser entendidos pela ProcITG. O método utilizado para a integração dos dois módulos é: fazer uma estrutura de dados idêntica à estrutura da CMDB, permitindo assim que os dados obtidos, a através de invocação a chamadas externas, possam ter um mapeando mecânico.

Os dados que passam entre as duas aplicações são os outputs das funções externas invocadas pela ProcITG durante a sua execução As funções externas que são invocadas pela ProcITG estão detalhadas na secção 6.XX.

Estão listadas as funções, cujos outputs são utilizados para a execução da ProcITG, e a sua correspondência na ProcITG. Os detalhes estão no dicionário de dados. Essas funções são as seguintes:

Quadro: Função CMDB out = dados ProcITG

- C_Relacao → sem correspondência
- L_CIs → devolve uma lista de 'Instancia'
- L_MetaCIs → devolve uma lista de 'CI'
- L_RelacoesTipo → devolve uma lista de estruturas do tipo 'CI_Destino_t'
- R_CI → devolve uma estrutura do tipo 'instanciaDetails_t'
- R_MetaCI → devolve uma estrutura do tipo 'MetaCI'
- R_Value → devolve um valor (texto)
- U_Value → sem correspondência
- U_AtribsCI → actualiza o valor de todos os atributos do CI em causa

ESPECIFICAÇÃO DA BASE DE DADOS

DIAGRAMA LÓGICO DO GESTOR DE ALTERAÇÕES

Através da entidades existentes na CMDB é possível criar as entidades do desenho lógico. Os 'RFC' são criados por um determinado 'Utilizador'. Este escolhe uma determinada 'Configuração', que é composta por um ou mais 'CI_'s. Sobre um determinado CI, o Utilizador pode realizar um conjunto predefinido de operações CRUD, desde que tenha as permissões associadas ao seu papel. Quando um RFC é criado, pode-se gerar a EFN. À medida que o RFC vai sofrendo alterações é possível gerar a EFN correspondente à nova versão do RFC.

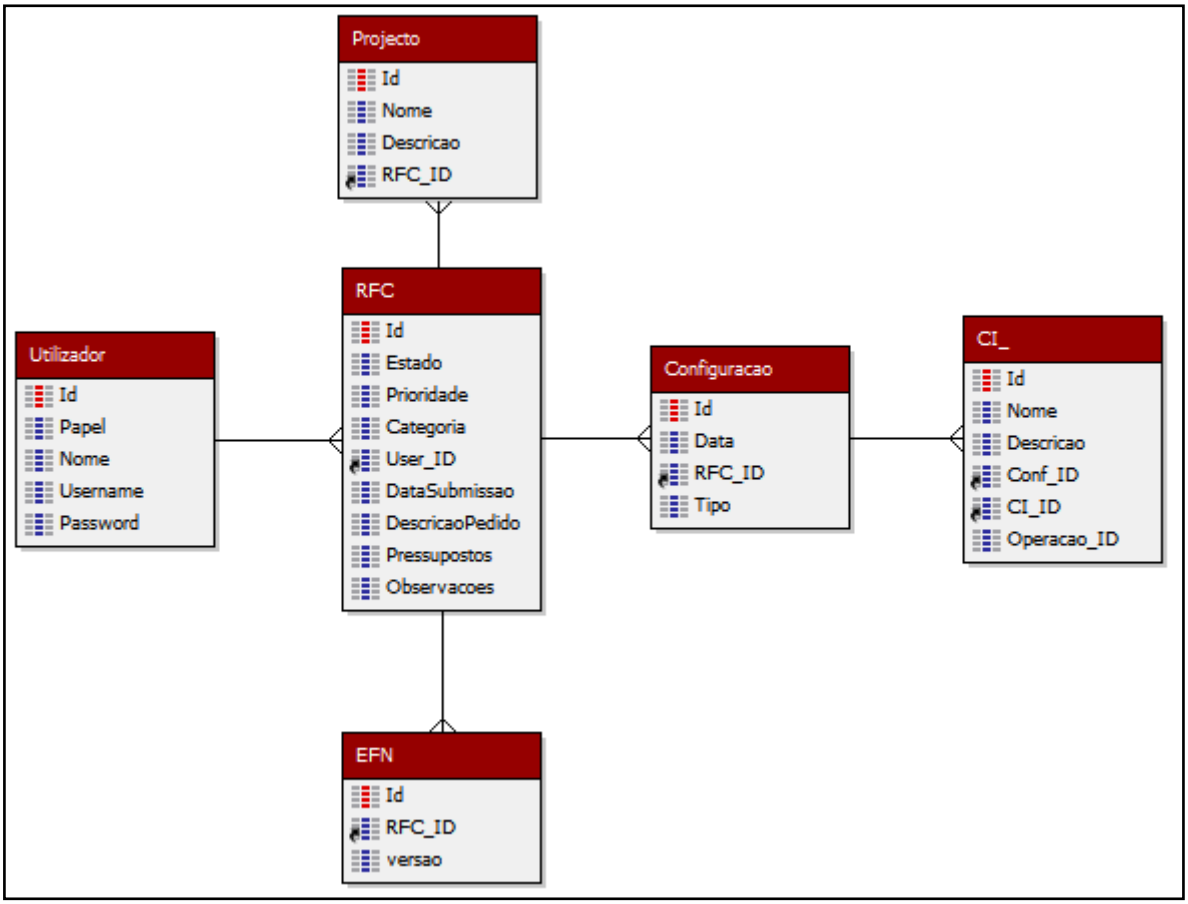
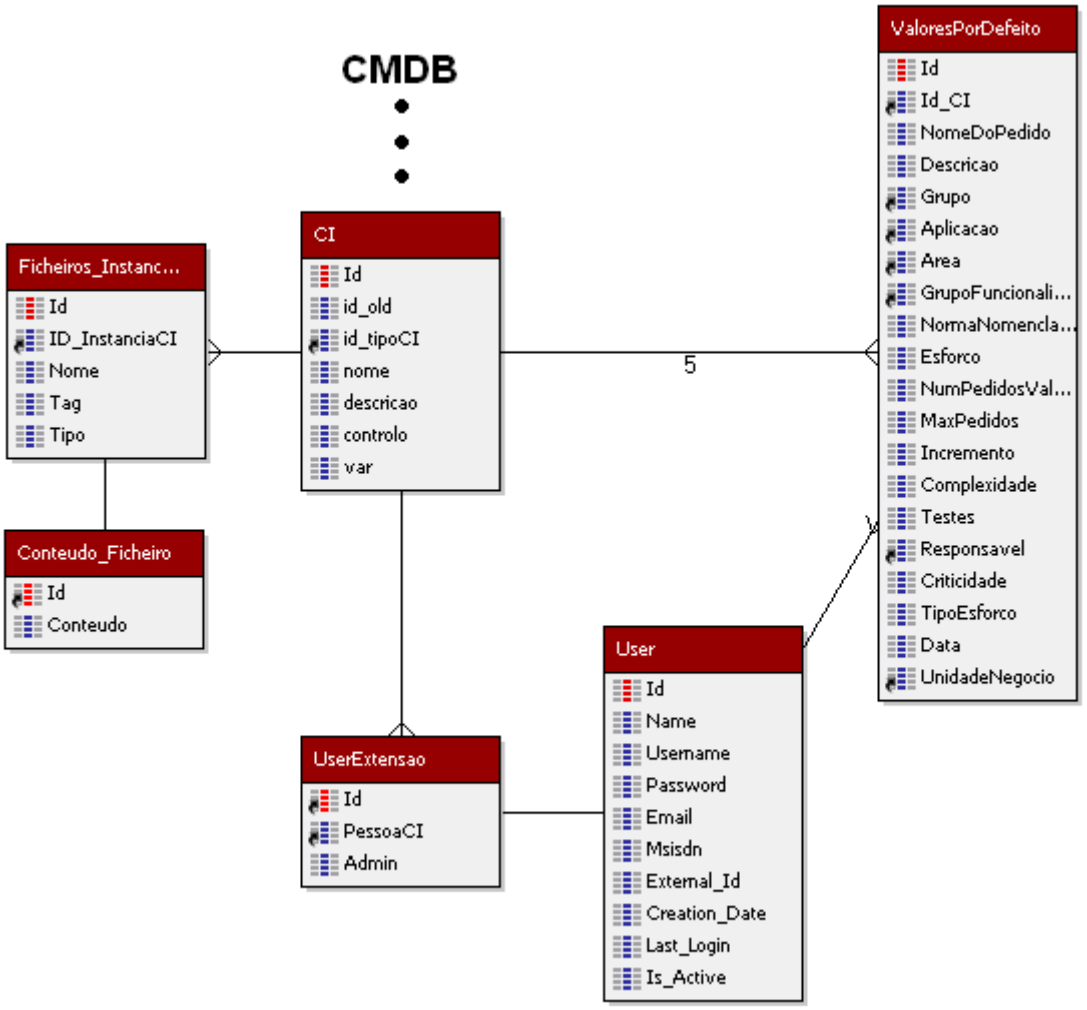


DIAGRAMA LÓGICO DO GESTOR DE TIPIFICADOS

Através da entidades existentes na CMDB é possível criar as entidades do desenho lógico. Os 'Pedidos Tipificados' são criados por um determinado 'Utilizador'. Este escolhe uma determinada 'Configuração', que é composta por um 'CI' e por uma entrada na tabela de 'Valores Por Defeito'. Sobre um determinado Pedido Tipificado, o Utilizador pode realizar um conjunto predefinido de operações de alteração de estado (Aprovado, Execucao, Fechado) e solicitação de documentos (orçamentos e

EFN guardados na tabela Ficheiros_InstanciaCI), desde que tenha as permissões associadas ao seu papel.



ESPECIFICAÇÃO DO MÓDULO GESTÃO DE ALTERAÇÕES

INTRODUÇÃO

A implementação deste sistema assenta em dois tipos de Web Flows:

- Internos, criados pelo programador para o sistema em questão;
- Externos, podem ser bibliotecas externas ou mesmo funções disponibilizadas por outros programas.

CONTEXTO

Diagrama de Contexto do Módulo

O diagrama deve identificar todos os fluxos de informação entre o subsistema, os subsistemas adjacentes e as interfaces externos. Ficheiros que sejam acedidos devem ser identificados.

Interfaces

As funções externas utilizadas, pertencem ao motor da CMDB, e são as seguintes:

Nome	Output
C_Relacao	
L_CIs	lista de 'Instancia'
L_MetaCIs	lista de 'CI'
R_CI	estrutura do tipo 'instanciaDetails_t'
U_Value	
UC_CI	

Tabela 7- Funções CMDB utilizadas

PROCESSAMENTO DO SUBSISTEMA

O programa é composto por web screens. O ponto de entrada da aplicação é um web screen denominado "WebScreen1". Este tem associados um conjunto de "Parâmetros" e de "Screen Actions", apresentados nas tabelas seguintes.

PARÂMETROS

Nome	Descrição	Obrigatório	Parametro	Tipo
id_CI	Identificação do CI; O seu preenchimento permite fazer a triagem dos dados a apresentar na página que vai ser carregada, juntamente com o parâmetro de entrada id_instancia.	No	Entrada	MetaCI
id_instancia	dentificação da instância; O seu preenchimento permite fazer a triagem dos dados a apresentar na página que vai ser carregada, juntamente com o parâmetro de entrada id_CI.	No	Entrada	CI
InFrame	Default	No	Entrada	Boolean
action	O seu preenchimento permite fazer a selecção da página que vai ser carregada.	No	Entrada	Text

SCREEN ACTIONS

Nome	Descrição
Preparation	N/A
Accao_Local	Esta função é chamada pelo botão "Ok" do campo "Escolher Instância". Permite criar a variável de sessão "arvore"; permite disponibilizar os componentes afectados.
Criar_RFC	Cria um RFC através dos dados inseridos pelo utilizador. Utiliza funções externas (dinamico)
EditarRFC	Função despoletadas pelo botão "Ok" presente no ecrã Editar. Guarda na CMDB os dados que o utilizador alterou.
GetEFN	Gerar a Especificação de Funcional de Alto Nível em html, a partir dos dados do RFC vindos da CMDB.

As Screen Actions estão descritas com maior detalhe na secção 6.7.

ACÇÕES LOCAIS E EXTERNAS

O sistema também utiliza um conjunto de acções que podem ser de origem local ou externa:

- Acções Locais – Acções criadas pelo programador;
- Acções Locais – Acções disponibilizadas por outras Bibliotecas.

Acções Locais

Get_arvore

Esta função é recursiva. Através de um identificador cria a arvore de dependentes, adicionando-os a uma lista denomida árvore.

Nome	Descrição	Sentido	Tipo	Obrigatório

In 1	Identificador do CI a partir do qual é criada a arvore descente.	Entr ada	CI	Yes
ni vel	Nível actual da árvore	Entr ada	Integer	Yes
list a	Lista de CI que vão ser adicionados à variável de sessão "arvore".	Entr ada	RecordList : ItemArvore	No
O ut1	Lista correspondente à variável de sessão "arvore".	Saí da	RecordList : ItemArvore	

Acções Externas

As acções externas utilizadas são da biblioteca WidgetLibrary40, e do motor CMDB (dinamico). Nomeadamente:

- WidgetLibrary40: Feedback_SetMessage, GetPhysicalPageName, ListNavigation_GetStartIndex, ListNavigation_ResetStartIndex;
- dinamico: D_MetaCI, D_MetaRelacao, L_CIs, L_MetaCIs, L_MetaRelacoes, L_Relacoes, L_RelacoesTipo, L_RelacoesTri, R_CI, R_MetaCI, R_MetaRelacao, R_Value, U_AtribsCI, U_Value, UC_CI, UC_MetaRelacao, UC_Relacao.

UTILIZAÇÃO DE DADOS

Os dados utilizados pelo sistema estão divididos em três grupos: entidades, estruturas, variáveis de sessão.

Estruturas

ItemArvore

Elemento da árvore de dependências

Atributo	Descrição	Tipo	Obrigatório
nivel	Hierarquia da árvore. O primeiro elemento tem nível zero.	Integer	No
alteracao	Operação que vai ser realizada sobre o item. Pode ser CRUD.	Integer	No
Id_CI	Identificador do CI	MetaCI	No
nomeConceito	Nome do conceito associado ao item	Text (100)	No
Id_instCI	Identificador da instancia de CI	CI	No
nome	Nome do item	Text (100)	No
descricao	Descrição do item	Text (400)	No

CONSIDERAÇÕES ESPECIAIS

Esta aplicação é directamente dependente da CMDB. Posto isto, quer a nível de performance, quer a nível de recuperação, a aplicação está dependente das mesmas características da CMDB.

ESPECIFICAÇÃO DE PROCEDIMENTOS

As especificações de procedimentos são as "Screen Actions" já referidas no início do capítulo. Os sub capítulos seguintes destinam-se à especificação das mesmas:

PREPARATION

É procedimento principal. Através do parâmetro de entrada "action" é desencadeado uma sequência de acções que leva a diferentes ecrãs destino, especificados no "WebScreen1". Os parâmetros de entrada são os mesmos do "WebSreen1", referenciados no sub-capítulo 7.3.1.

ACCAO_LOCAL

Esta função é chamada pelo botão "Ok" do campo "Escolher Instância". Permite criar a variável de sessão "arvore"; permite disponibilizar os componentes afectados.

Esta função tem dois parâmetros de entrada:

- id_instancia2: Identificador da Instancia sobre a qual vai ser criado um RFC;
- Opcao: Este parâmetro distingue se é realizada 1 ou as 2 finalidades da função.

CRIAR_RFC

Esta função cria um RFC através dos dados inseridos pelo utilizador, e utiliza funções do motor da CMDB (dinamico).

EDITAR_RFC

Esta função é despoletada pelo botão "Ok" presente no ecrã Editar, e faz o update dos dados que o utilizador alterou, directamente na CMDB, através de funções externas (UC_CI e U_Value) pertencentes ao "dinamico".

Esta função recebe como parâmetro de entrada, todos os dados do RFC.

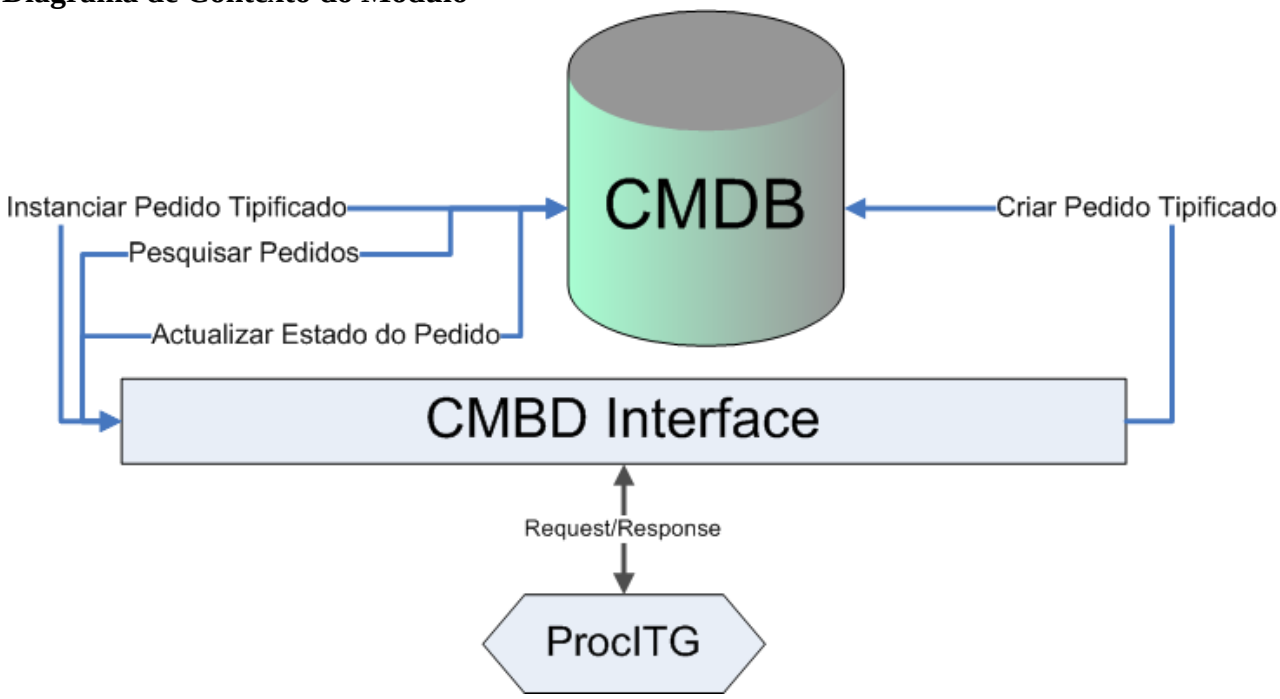
ESPECIFICAÇÃO DO MÓDULO DE GESTÃO DE PEDIDOS TIPIFICADOS

INTRODUÇÃO

Este módulo permite criar Pedidos Tipificados na CMDB através de uma interface mais simples e direccionada para o efeito. É possível manipular o estado de cada pedido e efectuar pesquisas através dos vários atributos de cada um. Adicionalmente, a aplicação permite gerar documentos úteis ao processo dos pedidos tipificados, como p. ex. Orçamentos e EFN.

CONTEXTO

Diagrama de Contexto do Módulo



Criar Pedido Tipificado:
From ProcITG → CMDB: MetaCI + Lista de MetaAtributos
Instanciar Pedido Tipificado:
From ProcITG → CMDB: CI + Lista de Atributos
Pesquisar Pedidos:
From ProcITG → CMDB: Ccritérios de Pesquisa
From CMDB → ProcITG: Lista de CIs
Alterar Estado do Pedido:
From ProcITG → CMDB: Identificador CI + Novo Valor de Estado

Interfaces
As funções externas utilizadas, pertencem ao motor da CMDB, e são as seguintes:

Nome	Output
C_Relacao	
L_CIs	lista de 'Instancia'
L_MetaCIs	lista de 'CI'

L_RelacoesTipo	lista de estruturas do tipo ‘CI_Destino_t’
R_CI	estrutura do tipo ‘instanciaDetails_t’
R_MetaCI	estrutura do tipo ‘MetaCI’
R_Value	valor (texto)
U_Value	
U_AtribsCI	

Tabela 8- Funções CMDB utilizadas

PROCESSAMENTO DO SUBSISTEMA

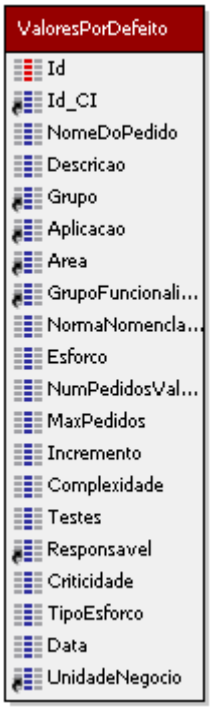
Screenshots e descrição das funcionalidades por ecrã.

UTILIZAÇÃO DE DADOS

Para além dos já referidos dados, tabelas e estruturas provinientes da CMDB, esta aplicação apnas utiliza/manipula as seguintes entidades informacionais:

Entidades:

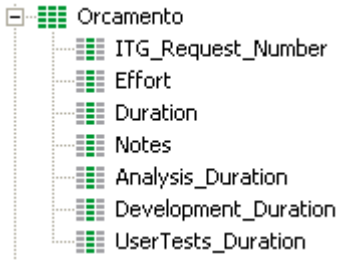
ValoresPorDefeito: Sempre que é criado um novo ‘Pedido Tipificado’, para além de dar entrada um registo na Tabela ‘MetaCI’ na CMDB, em simultâneo é criado um novo registo nesta tabela com os seguintes campos, a maioria preenchidos pelo utilizador:



Ficheiros_Instancia: Tabela onde são guardados os documentos de requisitos que o utilizador tem a possibilidade de guardar. Como Tabela auxiliar a esta, surge a Tabela Conteudo_Ficheiro criada por uma questão de boa metodologia de programação e que, na realidade, armazena o conteúdo binário dos ficheiros.
UserExtensao: Tabela intermédia que liga a, inalterável, Tabela User com a Tabela MetaCI da CMDB.

Estruturas:

Orcamento: Estrutura Auxiliar que permite ler a estrutura (Ficheiro Excel) de um ficheiro de orçamento:



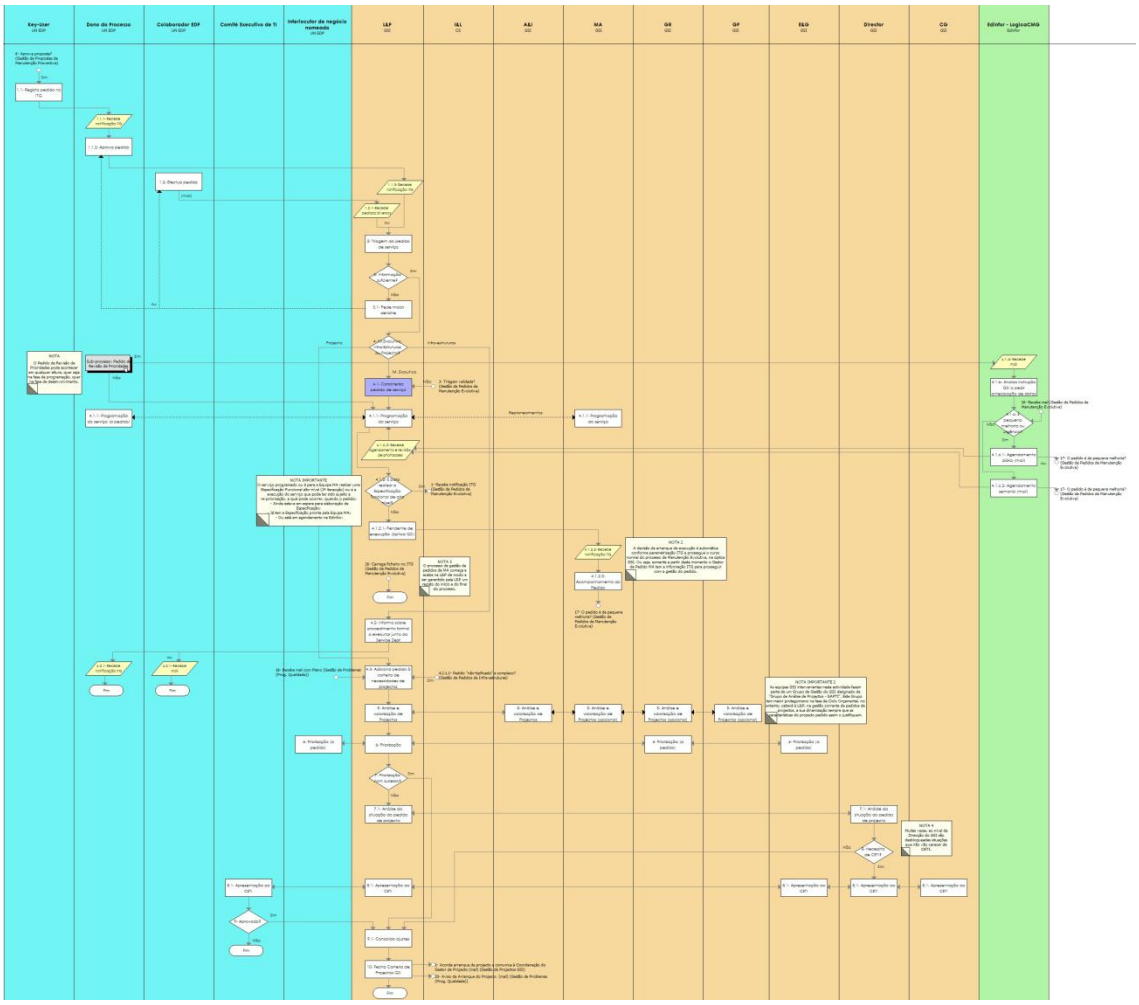
AtributosComValor: Estrutura Auxiliar que permite guardar um MetaAtributo (tipo) proviniente da CMDB e o seu futuro Valor.

CONSIDERAÇÕES ESPECIAIS

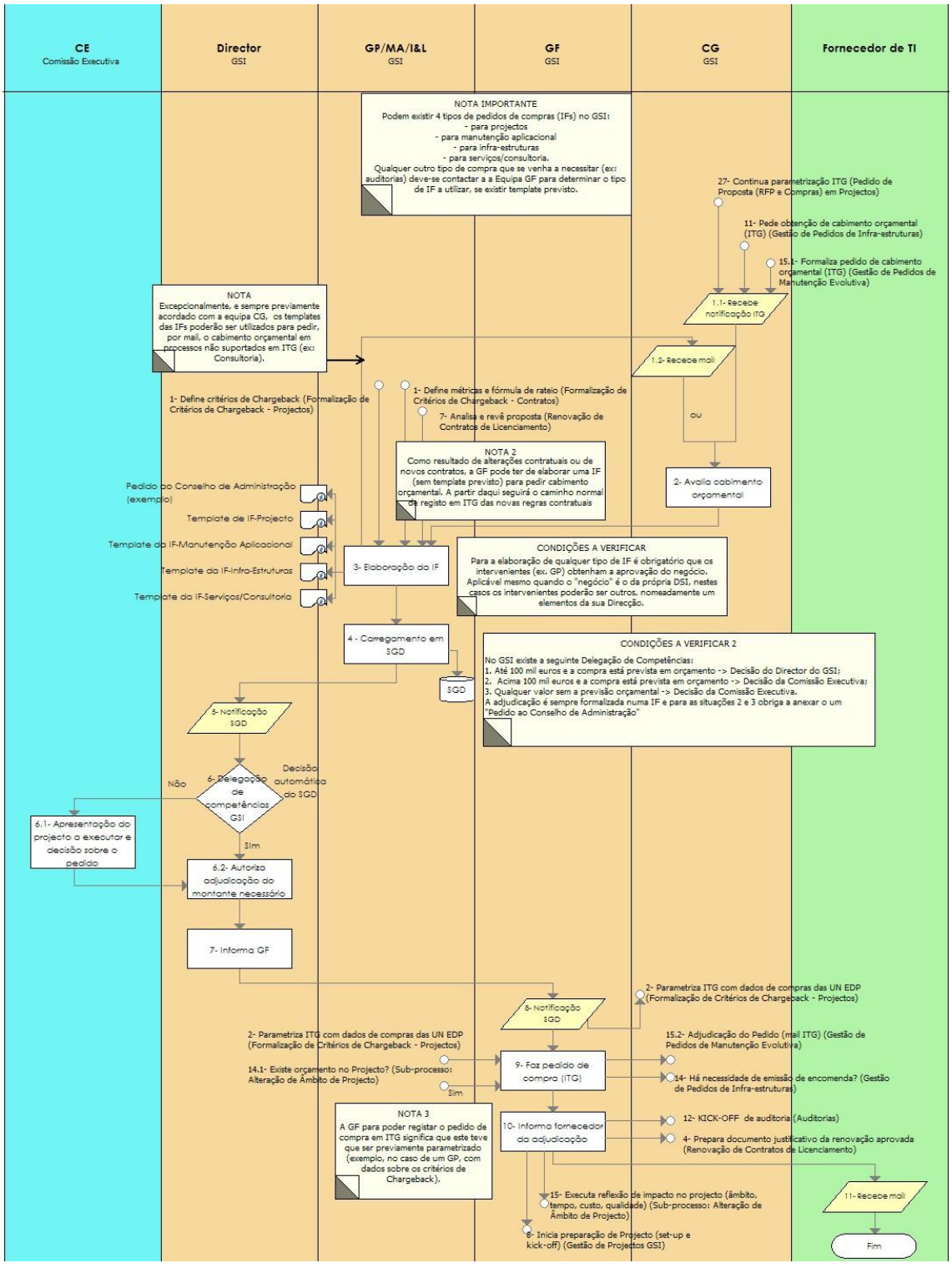
ProcITG: Esta aplicação é directamente dependente da CMDB. Posto isto, quer a nível de performance, quer a nível de recuperação, a aplicação está dependente das mesmas características da CMDB.

Anexo 3 – Processos Relacionados com a Manutenção Aplicacional

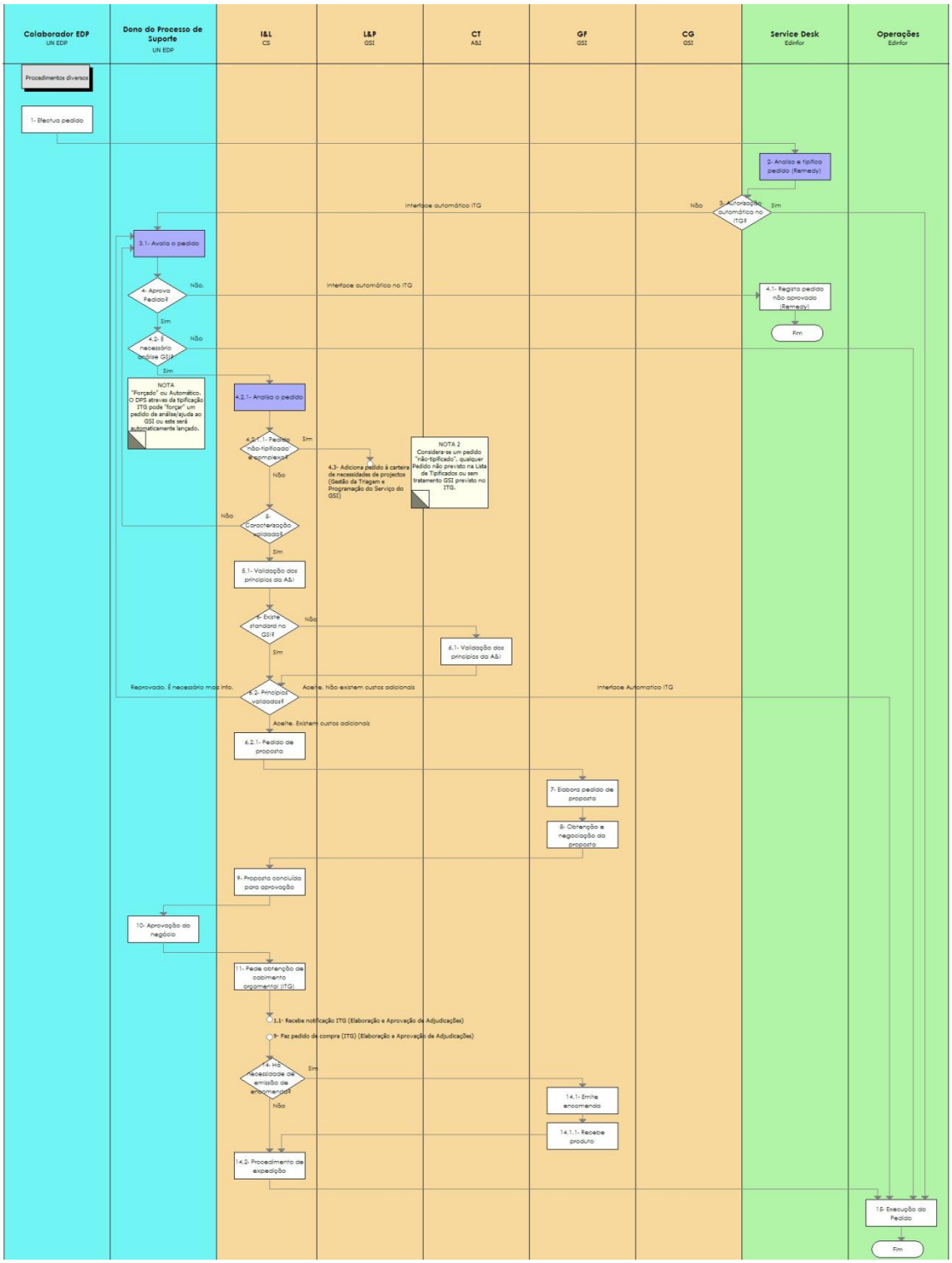
Processo Gestão da Triagem e Programação do Serviço



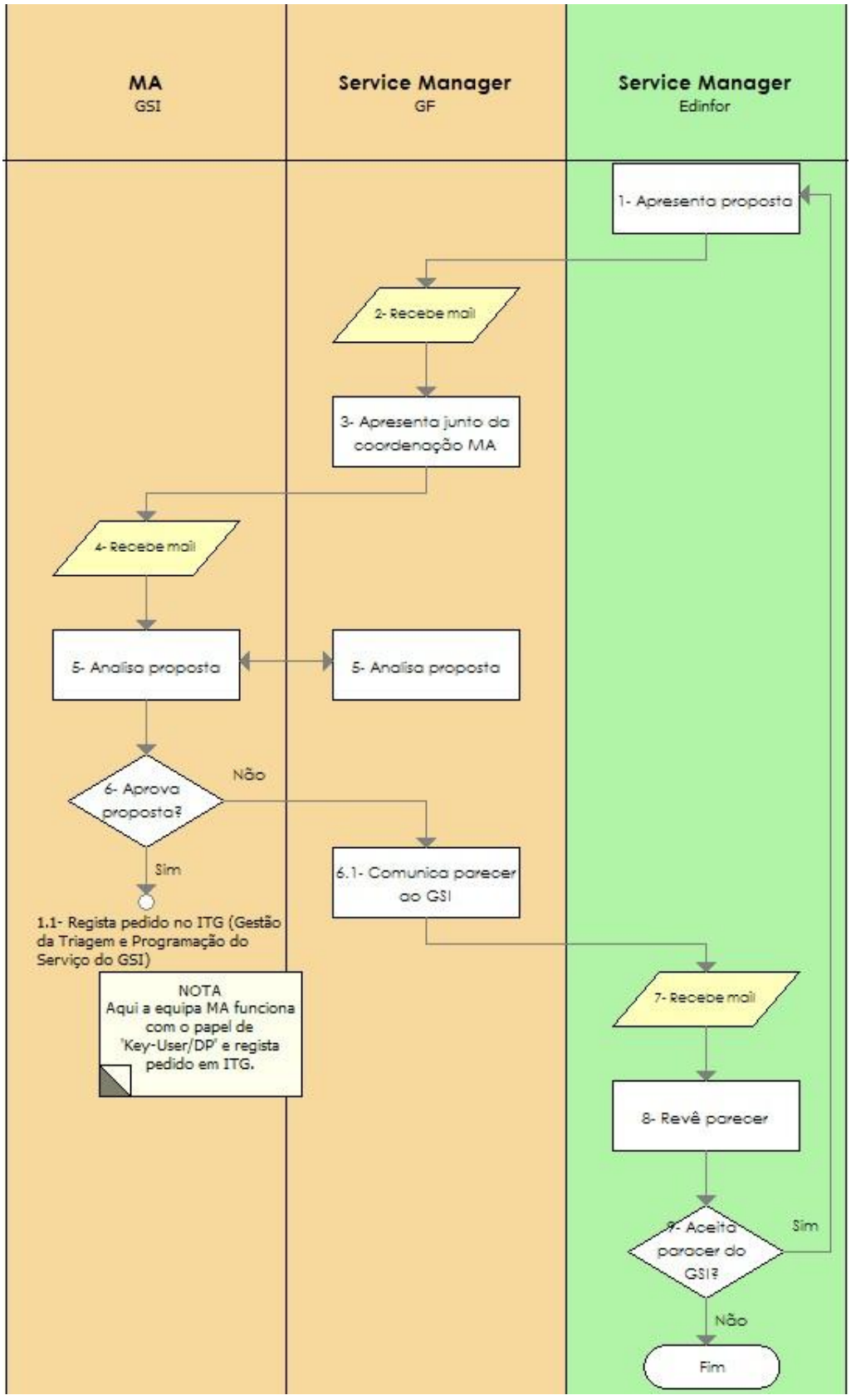
Processo Pedido de Revisão de Prioridades



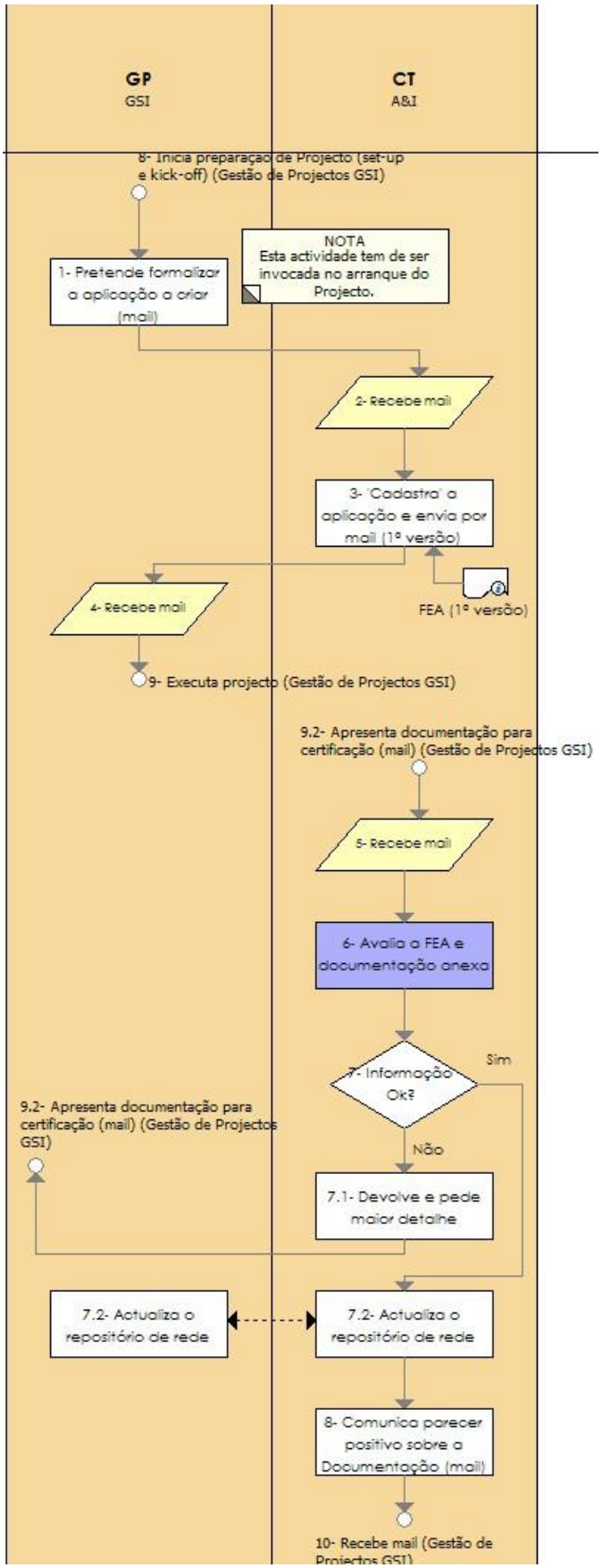
Processo Gestão de Pedidos de Infra-estrutura



Processo Gestão de Proposta de Manutenção Preventiva

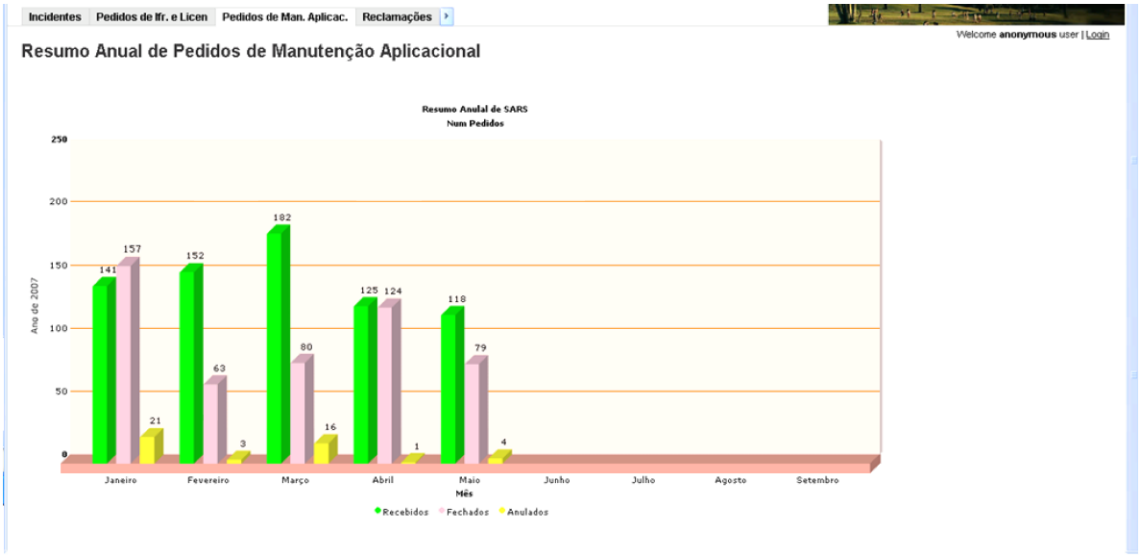


Processo Gestão Documental

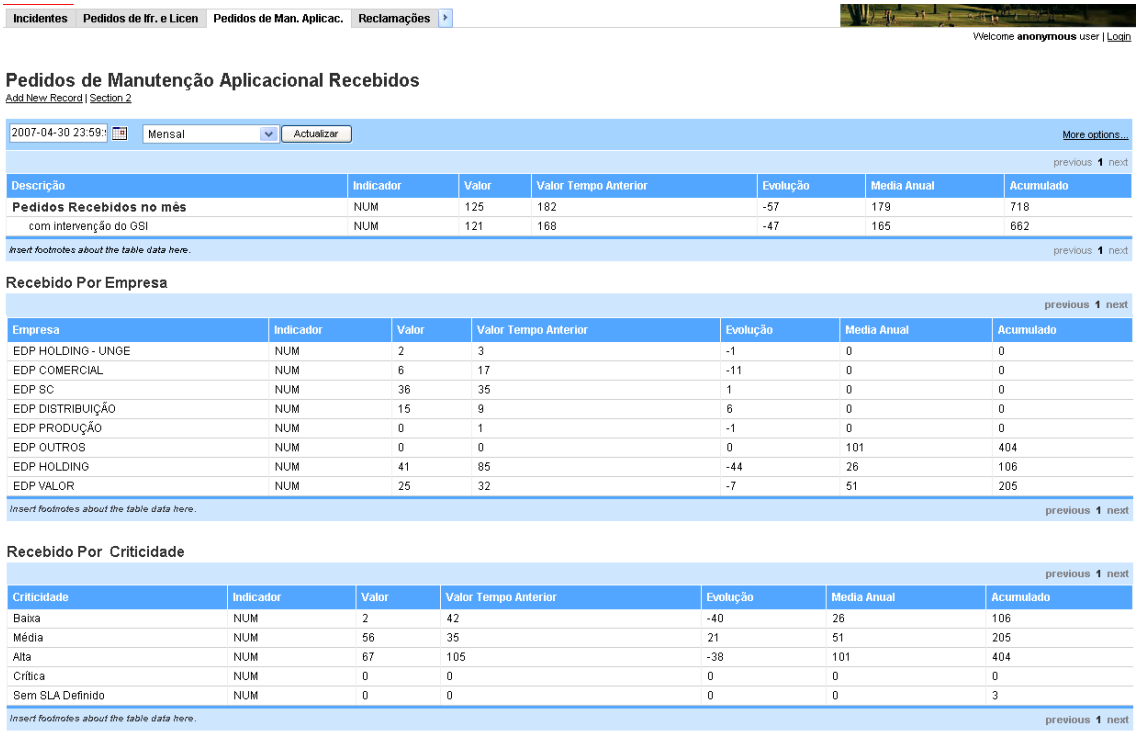


Anexo 4 – Pedidos da Manutenção Aplicacional

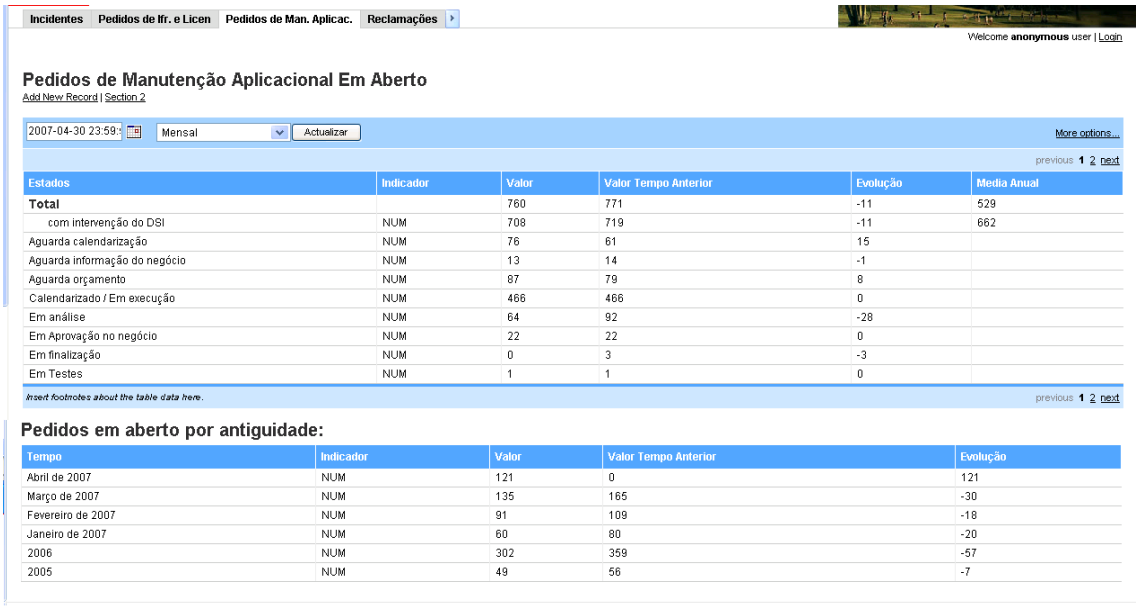
Resumo Anual de Pedidos de MA



Resumo Mensal dos Pedidos Recebidos de MA



Resumo Mensal dos Pedidos em Aberto de MA



Incidentes | Pedidos de Ifr. e Licen | Pedidos de Man. Aplicac. | Reclamações

Welcome **anonymous** user | Login

Pedidos de Manutenção Aplicacional Em Aberto

Add New Record | Section 2

2007-04-30 23:59:59 | Mensal | Actualizar | More options...

previous 1 2 next

Estados	Indicador	Valor	Valor Tempo Anterior	Evolução	Media Anual
Total		760	771	-11	529
com intervenção do DSI	NUM	708	719	-11	662
Aguarda calendarização	NUM	76	61	15	
Aguarda informação do negócio	NUM	13	14	-1	
Aguarda orçamento	NUM	87	79	8	
Calendarizado / Em execução	NUM	466	466	0	
Em análise	NUM	64	92	-28	
Em Aprovação no negócio	NUM	22	22	0	
Em finalização	NUM	0	3	-3	
Em Testes	NUM	1	1	0	

Insert footnotes about the table data here.

previous 1 2 next

Pedidos em aberto por antiguidade:

Tempo	Indicador	Valor	Valor Tempo Anterior	Evolução
Abril de 2007	NUM	121	0	121
Março de 2007	NUM	135	165	-30
Fevereiro de 2007	NUM	91	109	-18
Janeiro de 2007	NUM	60	80	-20
2006	NUM	302	359	-57
2005	NUM	49	56	-7