UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE INFORMÁTICA
CIÊNCIA DA COMPUTAÇÃO

MAITÊ FRIEDRICH DUPONT

Design e Projeto de um núcleo comum para desenvolvimento de sistemas web integrados a aplicativos móveis : G4 WORK

Monografia apresentada como requisito parcial para a obtenção do grau de Bacharel em Ciência da Computação.

Prof. Dr. Marcelo Soares Pimenta
Orientador

Porto Alegre, julho de 2014.

AGRADECIMENTOS

Agradeço aos meus pais pelas oportunidades que me proporcionaram;

Ao Matheus, por todo o apoio e conforto durante o curso de graduação e a escrita deste trabalho;

Ao professor e orientador Marcelo Pimenta pela orientação e dedicação;

À Universidade Federal do Rio Grande do Sul pelo ensino de excelente qualidade.

SUMÁRIO

AGRADECIMENTOS....3
LISTA DE ABREVIATURAS E SIGLAS....6
LISTA DE FIGURAS....7
LISTA DE TABELAS....8
RESUMO....9
ABSTRACT....10
1 INTRODUÇÃO....11
1.1 Objetivo....12
1.2 Estrutura do Trabalho....12
2 FUNDAMENTOS E CONCEITOS BÁSICOS....13
2.1 Dispositivo Móvel....13
2.2 Aplicativo....13
2.3 Aplicação Web....14
2.3.1 Servidor....15
2.3.2 Cliente....15
2.4 Integração....15
3 SOLUÇÕES JÁ EXISTENTES....16
3.1 Modo de Operação Atual....16
3.1.1 Banco de Dados....17
3.1.2 Aplicação Web....17
3.1.3 Integração....17
3.1.4 Aplicativos Móveis....18
3.1.4.1 uMov.me....18
3.1.4.2 zynk....19
3.1.4.3 G4 Apps....20
3.1.5 Principais Dificuldades....20
4 G4 WORK: "PROTO-FRAMEWORK"....21
4.1 Modo de Operação Proposto....21
4.2 Exploração....22
4.2.1 Sistemas Web....22
4.2.2 Módulo de Integração....25
4.2.3 Utilização das tecnologias....26
4.3 Construção da nova topologia – definições....26
4.3.1 Campos customizáveis....28
4.3.2 Parametrização....29
4.3.3 Formas de comunicação....30
4.3.4 Domínios....31
4.3.5 Filas....32
4.3.6 Integração com plataformas de mobilidade....34
4.3.7 Validação de formulários....34
4.3.7 Plugins jQuery....36
4.3.8 Flexibilização de formulários....37
5 DESENVOLVIMENTO....42
5.1 Tecnologias....42
5.2 Funcionalidades Visadas....43
5.2.1 Módulo Principal....45
5.2.1.1 Gerência de Usuário....45

5.2.1.1.1 Gerência de Usuários Administradores, Coordenadores e Back-Offices46
5.2.1.1.2 Gerência de Usuários Executores....47
5.2.1.1.2.1 Gerência de Calendários....48
5.2.1.2 Gerência de Equipes....49
5.2.1.3 Gerência de Clientes....51
5.2.2 MailService....52
5.2.3 Integração uMov.me....53
6 EXEMPLOS DE FLUXO....53
6.1 Criação de Usuário Executor....54
6.2 Recebimento de dados do uMov.me....56
7 CONCLUSÃO....58
REFERÊNCIAS....60
BIBLIOGRAFIA....60

LISTA DE ABREVIATURAS E SIGLAS

AJAX	Asynchronous JavaScript and XML
API	Application Programming Interface
DAO	Data Access Object
ERP	Enterprise Resource Planning
ES	Engenharia de Software
HTTP	HyperText Transfer Protocol
IDE	Integrated Development Environment
JSPX	JavaServer Pages XML compliant
MVC	Model View Controller
PDF	Portable Document Format
SQL	Structured Query Language
STS	Spring Source Toolkit
XML	eXtensible Markup Language
URL	Uniform Resource Locator

LISTA DE FIGURAS

Figura 1 Topologia atual das Soluções G4....16
Figura 2: Tela inicial do aplicativo uMov.me....18
Figura 3: Tela inicial do aplicativo Zynk....19
Figura 4 Mapa de Conceitos para o G4 SALES....22
Figura 5 Mapa de Conceitos para o G4 SERVICES....23
Figura 6 Mapa de Conceitos para o G4 LOGISTICS....23
Figura 7 Mapa de Conceitos para o G4 AGRO....24
Figura 8 Fluxo de funcionamento do módulo de integração (envio de dados)....25
Figura 9 Fluxo de funcionamento do módulo de integração (recuperação de dados)....25
Figura 10 Topologia proposta e aprovada para o G4WORK....27
Figura 11: Relacionamento entre as camadas do padrão MVC....28
Figura 12: Representação Entidade-Relacionamento de um parâmetro do sistema....29
Figura 13: Modelo de dados utilizado na comunicação entre módulos....30
Figura 14: Estrutura auxiliar para carregar dados entre o módulo principal e os módulos de integração....31
Figura 15: A estrutura representante de todos os domínios usados no sistema....32
Figura 16: Generalização de Mensagens para evitar duplicata de código e atributos....33
Figura 17: Composição genérica de um módulo de integração....34
Figura 18: Diagrama de sequência do fluxo de validação padrão implementado no G4 WORK 36
Figura 19: Antigo sistema de filtros dos produtos G4....38
Figura 20: Novo sistema de filtros dos produtos G4....38
Figura 21: Modelo ER de armazenamento de filtros dinâmicos....39
Figura 22: Representação do DTO que carrega os dados do filtro de Equipes para o controlador de Equipes....41
Figura 23: Tecnologias utilizadas no G4 WORK....43
Figura 24: Módulos objetos da parte de implementação deste trabalho....44
Figura 25: Representação dos dados de um usuário não Executor....46
Figura 26: Representação dos dados de um usuário Executor....48
Figura 27: Representação dos dados de um calendário....49
Figura 28: Edição dos membros de uma equipe....50
Figura 29: Representação dos dados de um Cliente....52
Figura 30: Envio de um Executor à plataforma uMov.me – Passos 1 a 3....55
Figura 31: Envio de um Executor à plataforma uMov.me – Passos 4 e 5....56
Figura 32: Envio de um Executor à plataforma uMov.me – Passos 6 e 7....56
Figura 33: Recebimento de dados do uMov.me – Passos 1 a 4....57
Figura 34: Recebimento de dados do uMov.me – Passo 5....58
Figura 35: Recebimento de dados do uMov.me – Passos 6 a 7....58

LISTA DE TABELAS

Tabela 1: Exemplos de parâmetros usados na flexibilização dos sistemas.....29
Tabela 2: Exemplo de possíveis Domínios cadastrados no sistema.....32
Tabela 3: Configuração via Banco de dados dos filtros de Equipe.....41

RESUMO

Reconhecidamente, hoje em dia, a Engenharia de Software é fundamental ao processo de desenvolvimento de software em pequenas, médias e grandes empresas. Independentemente do tamanho do projeto, mas especialmente se for de médio a grande porte, ela traz inúmeros benefícios, entre eles o reúso de componentes, que torna o processo de desenvolvimento mais ágil e menos custoso. Uma das formas, hoje, mais comuns de reúso são os frameworks, onde códigos comuns a diferentes aplicações são reunidos de forma a criar um "repositório" de funcionalidades que podem ser genéricas ou específicas a um domínio.

Este trabalho propõe o projeto e início de implementação de um "proto-framework", para o domínio de aplicações da empresa X (a fim de manter a confidencialidade da empresa), tendo como base para análise as 4 soluções de software principais da empresa.

Primeiramente, o contexto de aplicação do framework é resumido, seguido de esclarecimentos sobre o domínio e comparativos entre as soluções. Posteriormente, são exibidos detalhes do projeto e da implementação realizada.

Palavras-chave: Engenharia de Software, reúso, sistemas web, framework.

ABSTRACT

Admittedly, nowadays, Software Engineer is fundamental to the software development process in small, medium and large companies. Independently of the size of the project, but especially if it's medium or large, it brings numerous benefits, for instance component reuse, that makes the developing process more agile and less expensive. Today, one of the most common ways to reuse components is through frameworks, where code similar to different applications are put together to create a repository of specific or generic functions to a domain.

This work propose the project and begin of implementation of a “proto-framework” for the company X it solutions' applications domain. It it is based on the analysis of the company's four main software solutions.

Initially, the application context of the framework is presented, followed by clarifications about the domain and a comparative of the solutions. After, project details and of the implementation are shown.

Keywords: Software Engineering, reuse, web systems, frameworks.

1 INTRODUÇÃO

A X é uma empresa criada em 2007 com o objetivo de prover soluções de mobilidade. Desde então a empresa formou parcerias com as plataformas de mobilidade uMov.me e Zynk, sendo a primeira para coleta de dados em campo (fora da sede física da empresa) e a segunda para envio de documentos para campo. Fica com a X, portanto, a tarefa de computar, armazenar e exibir estes dados de forma simples e objetiva, melhorando os tempos de resposta e a tomada de decisões dos clientes, e otimizando os seus processos de negócios.

Desde a sua criação até o ano de 2013, a X trabalhou com projetos voltados para cada cliente, desenvolvendo sistemas completos e sob medida para cada um. Em 2013, o foco da empresa passou a ser produtos genéricos com adaptações, ou seja, um sistema que é o mesmo para todos, mas são realizados ajustes para melhor adequação a cada cliente.

A partir desta nova modalidade foram criados as quatro soluções da X:

- X VENDAS para Equipes de Vendas
- X SERVIÇOS para Serviços Técnicos & Atendimentos
- X LOGÍSTICA para Automação de Logística
- X AGRO para gestão de Agronegócio

Cada um dos sistemas foi concebido independentemente, por equipes distintas, sendo que o X VENDAS e o X SERVIÇOS hoje já estão sendo comercializados, enquanto os outros dois estão em fase de planejamento.

Durante a concepção dos dois primeiros, percebeu-se uma grande semelhança em várias partes dos sistemas. As mais notáveis nas partes de integração com as plataformas de mobilidade, mas também no cadastro de usuários, gerenciamento de equipes etc.

Surgiu então a ideia de criar um núcleo comum de funcionalidades para as soluções X, de forma a aumentar o reúso e diminuir o tempo de desenvolvimento das próximas duas soluções (X LOGÍSTICA e X AGRO).

1.1 Objetivo

Este trabalho tem como objetivo propor à empresa X a utilização de um proto-framework (núcleo base de funcionalidades) específico às suas soluções, de forma a reduzir significativamente o tempo de desenvolvimento das soluções e reteste das funcionalidades já estabelecidas. O resultado do trabalho recebeu a denominação de “proto-framework”, pois é apenas o primeiro passo no desenvolvimento de um framework sólido para embasar os produtos da empresa.

Este proto-framework visa a padronização, unificação e otimização de códigos e fluxos com mesmas finalidades ou semelhantes, redução de códigos duplicados através da generalização de funções e o refinamento do modo de utilização de tecnologias já empregadas na empresa, a fim de tirar o maior proveito possível destas.

1.2 Estrutura do Trabalho

Neste capítulo 1 foi descrita a contextualização e motivação deste trabalho, bem como seu objetivo.

No segundo capítulo serão descritos conceitos básicos para o entendimento deste trabalho, assim como sua interação neste contexto.

O capítulo 3 introduz brevemente o modo de operação atual das soluções e tecnologias utilizadas nas mesmas.

No capítulo 4 é detalhado o projeto do conjunto de funcionalidades proposto e a arquitetura resultante.

O quinto capítulo detalha etapas do processo de desenvolvimento do trabalho.

O sexto capítulo exemplifica dois fluxos completos de funcionamento dos módulos.

O fechamento do trabalho, bem como conclusões e sugestões para aprimoramento do mesmo estão no capítulo 7.

2 FUNDAMENTOS E CONCEITOS BÁSICOS

Neste capítulo serão apresentados fundamentos e conceitos relevantes para a compreensão deste trabalho. Serão resumidos os conceitos de Aplicativo, Aplicação Web, Dispositivo Móvel, Integração, Cliente, Servidor e outros.

2.1 Dispositivo Móvel

O termo “Dispositivo Móvel” pode designar qualquer tipo de computador de bolso, o que geralmente inclui Smartphones, PDAs, Notebooks, Tablets etc. Neste trabalho, o termo fica restrito a Smartphones e Tablets, dispositivos estes onde podem ser instalados os aplicativos utilizados pela X em suas soluções de mobilidade.

A vantagem de se utilizar dispositivos móveis em detrimento das antigas guias de papel é notável quando se ressalta o fato de que os dispositivos podem estar conectados à rede de dados. Esta conexão permanente torna obsoleto o processo de entrada dos dados vindos de campo na sede da empresa, já que todos os dados inseridos no dispositivo durante um dia inteiro de trabalho estão disponíveis em tempo real para a empresa.

Outras vantagens incluem a possibilidade de reprogramar ao longo do dia as agendas dos funcionários conforme a quantidade e agilidade dos atendimentos e a captura de informações sobre a localização do funcionário.

2.2 Aplicativo

Em geral, “Aplicativo” pode se referir a qualquer programa de computador que visa o desempenho de uma tarefa específica. Neste trabalho, serão referidos como “Aplicativos” apenas a subcategoria de “Aplicativos de Celular”, aqueles programas desenvolvidos especificamente para serem utilizados em Smartphones e Tablets.

A possibilidade de instalar em dispositivos móveis aplicativos desenvolvidos pela X ou seus parceiros permite ajustes finos na adaptação dos sistemas às necessidades de um cliente específico e a realização de tarefas de maneira sucinta e objetiva.

2.3 Aplicação Web

Aplicação Web designa um software projetado para ser utilizado através de um navegador (browser) com acesso a alguma rede (em geral a internet). Este software recebe requisições do browser – geralmente através do protocolo HTTP - realiza algum tipo de processamento (pode ser de qualquer natureza, incluindo acesso a bancos de dados, aplicações legadas etc), e devolve o resultado, em geral na forma de uma página da Web.

Comumente, as aplicações web são baseadas na arquitetura cliente-servidor, onde o cliente entra com os dados e o servidor os processa e armazena. Elas se popularizaram devido à ubiquidade dos browsers(clientes), que estão disponíveis para todos os sistemas operacionais hoje em dia. A ampla disseminação deste tipo de software garante que os desenvolvedores de uma aplicação web possam focar seus esforços no desenvolvimento do sistema em si, e não na criação de um cliente para cada tipo de computador e cada tipo de sistema operacional.

Ao contrário das aplicações nativas – de uma camada - que rodam isoladas em uma máquina, as aplicações web podem possuir um número qualquer de camadas, sendo três o número mais comum (PETERSEN, 2007). As camadas são: apresentação, aplicação e armazenamento. A camada de apresentação (implementada no navegador) exibe informações para o usuário, possibilita entrada de dados etc; a camada de aplicação é a responsável pelo recebimento das requisições vindas da apresentação, realização do processamento requisitado através das regras de negócio implementadas e retorno do resultado; a terceira e última camada, a de armazenamento, é responsável pela parte do processamento que envolve consulta e armazenamento de dados gerados e/ou recebidos pela aplicação.

O desenvolvimento de aplicações web é frequentemente facilitado pelo uso de componentes previamente desenvolvidos com esta finalidade chamados frameworks de aplicações web. Estes componentes possibilitam o rápido desenvolvimento provendo as diversas funcionalidades que são comuns a todas as aplicações deste tipo, como gerência de acesso e persistência de dados (WEB APPLICATION, 2010).

2.3.1 Servidor

A máquina (pode ser uma máquina virtual) acessada através de uma rede que recebe requisições, realiza processamentos e retorna resultados. Em aplicações Web geralmente é acessada através do protocolo HTTP, e retorna páginas HTML. É onde fica armazenada a Aplicação Web, e por isso deve garantir todos os requisitos de performance e disponibilidade da mesma.

2.3.2 Cliente

O software que realiza requisições para o Servidor, em aplicações Web geralmente pelo protocolo HTTP. Pode ser um navegador (no caso de uma aplicação web), ou um software de simulação de requisições HTTP, ou mesmo outro sistema qualquer.

2.4 Integração

Integração, para este trabalho, é o processo de transferir dados entre uma Aplicação Web e outros sistemas. Esta transferência, no caso de aplicativos móveis, pode ser em ambos ou em apenas um sentido, mas é sempre de maneira ativa (do ponto de vista da aplicação web). A comunicação entre a aplicação e os aplicativos das plataformas de mobilidade uMov.me e Zynk se dá através de uma API exposta por estas plataformas, enquanto a comunicação com aplicativos próprios da X acontece via um banco de dados intermediário.

Em alguns casos, pode haver integração entre a Aplicação Web e outros sistemas do cliente, como ERPs e outros bancos de dados. Esta integração pode acontecer através de consultas aos bancos de dados gerenciados por estes sistemas, API ou troca de arquivos, onde cada um dos sistemas lê e escreve arquivos de layout pré-definido em um local determinado.

3 SOLUÇÕES JÁ EXISTENTES

Aqui serão descritas as tecnologias empregadas hoje no desenvolvimento das soluções X, suas características e seu funcionamento em conjunto.

3.1 Modo de Operação Atual

Hoje, a empresa trabalha com sistemas modulares, mas a granularidade dos módulos não permite o reúso dos mesmos. Como vimos no capítulo 1, a X possui quatro soluções de mobilidade, e todas seguem a mesma topologia, que pode ser vista na Figura 1:

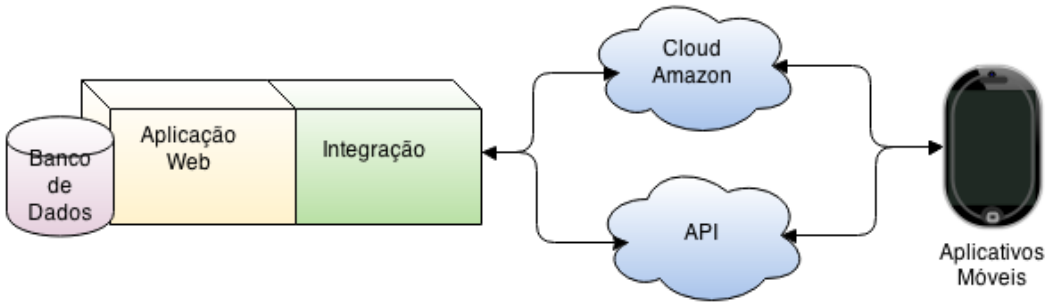

Figura 1 Topologia atual das Soluções X

Como é visível na Figura 1, cada solução é composta por dois módulos principais (Aplicação Web e Integração) que compartilham informações através de um banco de dados. As informações são compartilhadas com aplicativos móveis disponíveis em dispositivos de mobilidade de duas formas possíveis: através de um banco de dados em nuvem, no caso de aplicações da X e através de APIs no caso das plataformas uMov.me e Zynk.

Para cada cliente, é replicada a base deste sistema como um todo, e em cima deste modelo genérico são feitos os ajustes necessários.

3.1.1 Banco de Dados

O banco de dados serve como intermediário entre o módulo rodando a aplicação Web e o módulo de integração. Para isto, ambos precisam conhecer todos os conceitos implementados no sistema. Todo o banco de dados é acessado pelos dois módulos, gerando mais concorrência e, consequentemente, diminuindo a performance do sistema, além de replicação de regras de negócio nos dois módulos para garantir a correção dos dados no banco.

O modelo de banco de dados utilizado é o modelo Relacional, implementando os conceitos na forma de tabelas, linhas e colunas, ligadas por relacionamentos.

3.1.2 Aplicação Web

Este módulo é o responsável pela Aplicação Web, onde os usuários do Sistema Web têm acesso aos dados que estão inseridos na base de dados. O sistema possibilita o cadastro, alteração, e visualização destes dados de forma amigável e eficiente, gerando gráficos, estatísticas e comparações. Para ter acesso a este sistema, o usuário necessita ter sido previamente cadastrado e efetuar login.

3.1.3 Integração

O módulo de integração é responsável por enviar para as plataformas de mobilidade todos os dados que serão necessários para realização das atividades propostas nos mesmos e receber de volta os dados resultantes destas atividades. O envio de informações acontece de forma passiva, isto é, o envio de determinadas informações é requisitado pelo módulo Web através de uma chamada HTTP. Já o recebimento de informações acontece de forma ativa, onde o próprio módulo invoca métodos das plataformas de mobilidade para buscar os dados inseridos nas mesmas.

Além disso, também são responsabilidades do módulo de integração a geração de documentos e e-mails a pedido ou não do módulo Aplicação Web.

3.1.4 Aplicativos Móveis

Serão resumidos nesta sessão as plataformas de mobilidade utilizadas nas soluções X e seus respectivos aplicativos.

3.1.4.1 uMov.me

O uMov.me é uma plataforma de desenvolvimento de aplicativos móveis que possibilita a qualquer usuário a criação de um aplicativo customizado de forma rápida e simples. O usuário define as tarefas que serão realizadas no dispositivo móvel, quais campos e em que ordem serão exibidos etc. Para cada aplicativo criado, também é criado um portal nos servidores da uMov.me, onde os dados informados no aplicativo móvel ficam disponíveis para consulta. Os dados também podem ser consultados via API, como é feito no caso das soluções X.

Figura 2: Tela inicial do aplicativo uMov.me

3.1.4.2 zynk

O zynk é uma plataforma de distribuição de conteúdo para aplicativos móveis. Através dele é possível cadastrar grupos de usuários e enviar para grupos distintos conteúdos distintos. Os tipos de conteúdo passíveis de envio incluem PDFs, imagens, vídeos, arquivos de texto e outros. No momento, o Zynk não permite a distribuição de conteúdo de dispositivo móvel para dispositivo móvel, sendo necessário acessar o portal web da plataforma para fazer isto. Nenhum dado é buscado no Zynk pelas soluções X, eles são apenas enviados para a plataforma, que distribui os conteúdos para os usuários apropriados.

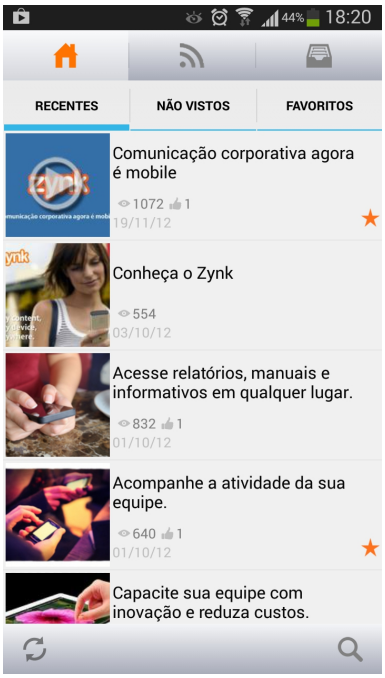

Figura 3: Tela inicial do aplicativo Zynk

3.1.4.3 X Apps

O conjunto de aplicativos chamado de X Apps é composto por aplicativos criados pela X para funções pontuais a serem realizadas nos aplicativos móveis como por exemplo, registrar a posição do usuário via GPS, limitar/bloquear a utilização de certas funcionalidades do dispositivo, exibir informações da solução X relacionada etc. Cada aplicativo é instalado independentemente dos outros e sob demanda do cliente.

A comunicação com estes aplicativos é feita via um banco de dados hospedado em uma nuvem da Amazon, de forma a garantir a maior disponibilidade possível para integração com os aplicativos móveis. Assim, mesmo que o sistema web fique fora do ar, os dados relacionados às atividades continuam a ser armazenados.

3.1.5 Principais Dificuldades

Apesar de haver um modelo genérico de cada sistema, que é clonado para cada cliente, no final da customização às vezes tem-se um sistema bastante diferente do original. Assim, um dos maiores problemas experimentados acontece quando um cliente solicita uma customização que a empresa julga relevante para os outros clientes. Neste caso, a alteração tem de ser replicada para cada cliente manualmente, podendo causar falhas inesperadas, já que o comportamento dos sistemas não é mais exatamente o mesmo.

Outro problema é a modificação de funcionalidades implementadas para todos os sistemas e para todos os clientes, que é feita diretamente em código, muitas vezes tendo efeitos colaterais graves em outras partes do sistema. Por exemplo: A mesma validação dos dados inseridos em formulários é feita de diversas maneiras pelo sistema, dentre elas validação campo a campo com envio de cada campo ao servidor, validação via javascript, e validação de todos os dados (enviados de uma só vez) no servidor. Ao inserir um campo na entidade Cliente tanto para o X SERVIÇOS quanto para o X VENDAS, as validações deste campo devem ser feitas em locais diferentes para cada sistema e cliente, mesmo que o comportamento da entidade seja o mesmo em todos eles, e o campo em si tem de ser inserido manualmente tanto na tela de criação quanto na de edição.

A falta de documentação oficial adequada e atualizada é um problema que agrava os já

citados pois dificulta imensamente a análise de impacto e esforço necessário e a rastreabilidade dos componentes.

4 G4 WORK: “PROTO-FRAMEWORK”

Aqui será detalhado o modo de operação proposto neste trabalho para o proto-framework G4 WORK, bem como os passos e métodos utilizados na fase de projeto para atingir este objetivo. Nesta fase foram realizadas análises mais específicas sobre os dois sistemas já desenvolvidos (X SERVIÇOS e X VENDAS) a partir das quais deriva um novo modelo de organização para os próximos sistemas.

4.1 Modo de Operação Proposto

Utilização do núcleo de funcionalidades G4 WORK desenvolvido aqui, juntamente à documentação do mesmo na base de conhecimentos (KB) Confluence – já utilizada pela empresa – e do melhor aproveitamento das ferramentas e tecnologias empregadas, para agilizar o processo de desenvolvimento e customização das soluções, o teste de novas funcionalidades e teste do sistema como um todo.

Com esta base comum a todas as soluções e clientes, uma nova funcionalidade relacionada às partes do sistema implementadas pelo projeto pode ser facilmente incluída em todos os produtos em desenvolvimento ou em produção de forma rápida e sem efeitos colaterais.

O projeto deste proto-framework foi realizado em duas fases: Exploração e Construção da Nova Topologia. Nenhuma metodologia específica para desenvolvimento de frameworks foi utilizada pois o trabalho foi bastante exploratório já que não existia uma lista de requisitos a serem cumpridos pelo sistema.

4.2 Exploração

A fase de exploração inclui três etapas que surgiram naturalmente durante o período exploratório: a exploração dos sistemas web, a exploração do módulo de integração com as plataformas de mobilidade e a exploração da utilização de tecnologias já utilizadas pela empresa.

4.2.1 Sistemas Web

Primeiramente, foi desenhado um modelo básico das principais entidades presentes nos sistemas existentes (X VENDAS e X SERVIÇOS) e como seriam as entidades nos outros dois sistemas (X LOGÍSTICA e X AGRO), o que foi chamado de Mapa de Conceitos. Os mapas gerados nesta etapa estão listados abaixo onde, em verde foram apontadas as partes em comum em todos os sistemas, e em branco ficaram as partes específicas a cada um.

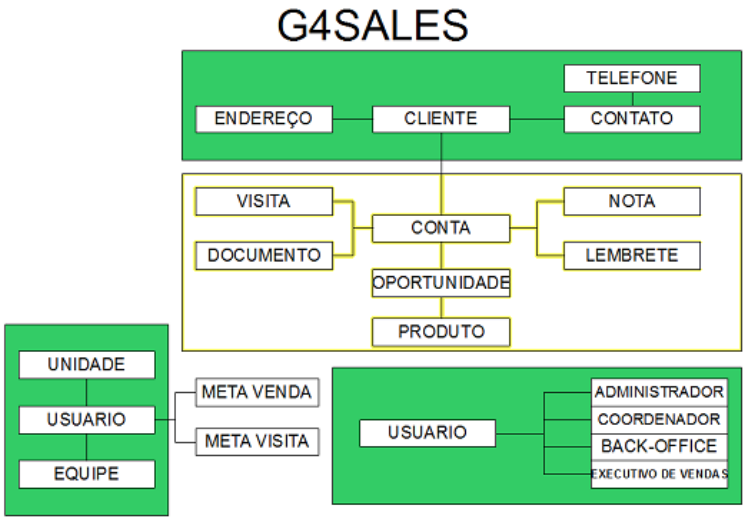

Figura 4 Mapa de Conceitos para o X VENDAS

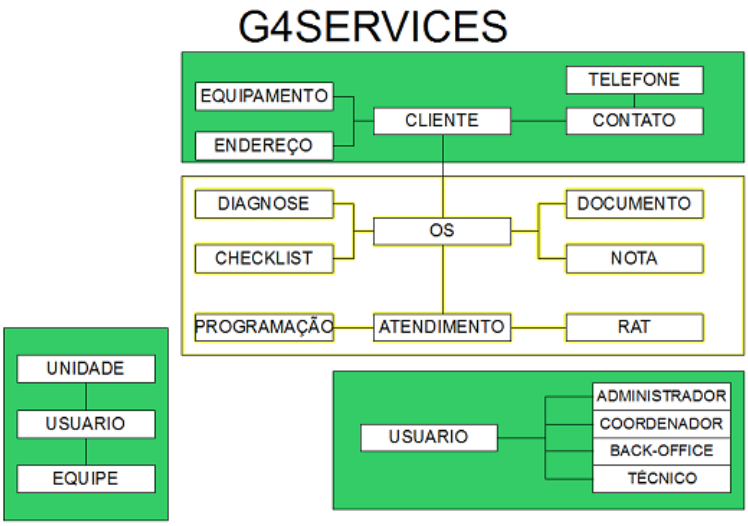

Figura 5 Mapa de Conceitos para o X SERVIÇOS

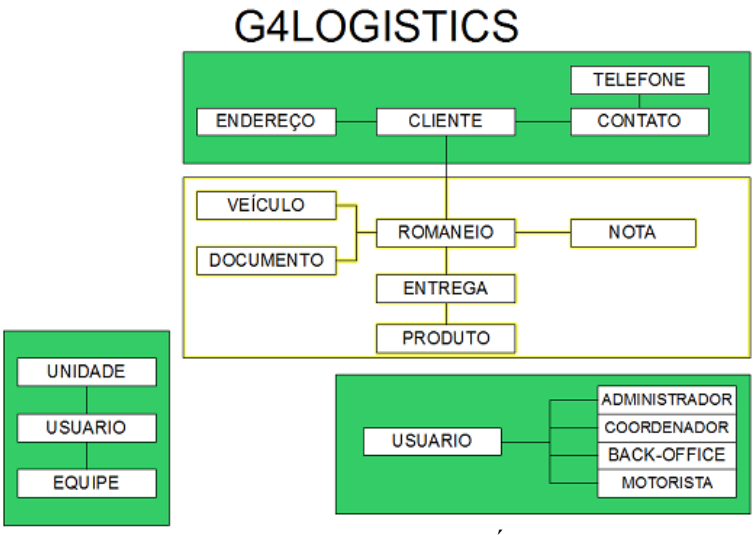

Figura 6 Mapa de Conceitos para o X LOGÍSTICA

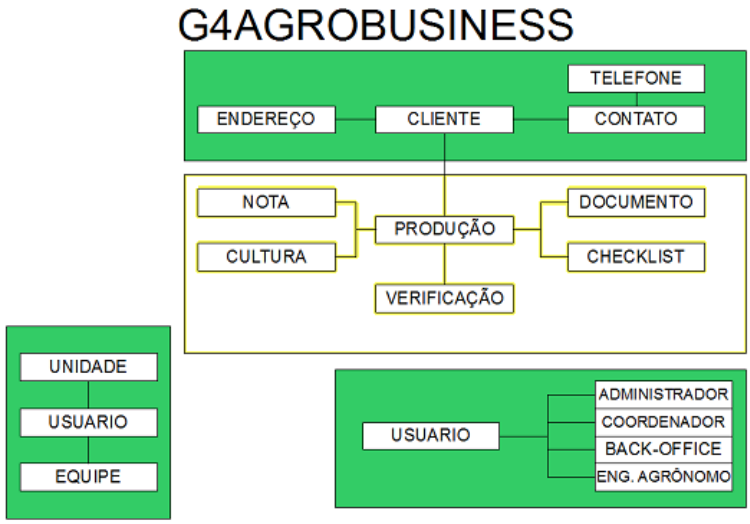

Figura 7 Mapa de Conceitos para o X AGRO

É possível perceber, a partir da organização dada aos conceitos e das cores utilizadas, que as semelhanças entre os sistemas são grandes, sujeitas apenas a algumas particularidades. Ao discutir o assunto com o Projetista da X, a conclusão foi de que as informações para estes conceitos ressaltados em verde (Usuário, Cliente, Equipe, Unidade, Endereço, Telefone e Contato) são, geralmente, as mesmas, com modificações pontuais. Sendo assim, estas partes comuns serão trabalhadas no G4 WORK até que sejam genéricas o suficiente para que o mesmo desenvolvimento possa ser utilizado para qualquer uma das soluções.

Para todas as soluções existem quatro tipos de Usuários: Administrador, que gerencia e supervisiona o sistema; Coordenador, que é líder de uma ou mais equipes, sendo responsável pelo desempenho da(s) sua(s) equipe(s); Back-Office, quem mais utiliza o sistema na sede (ou filial) da empresa contratante do sistema; o último perfil, que representa o usuário que de fato sai da empresa para realizar os atendimentos na rua, através de um dispositivo mobile. Este perfil foi generalizado como Executor.

4.2.2 Módulo de Integração

O comportamento do módulo de integração para envio de dados para as plataformas de mobilidade é como segue:

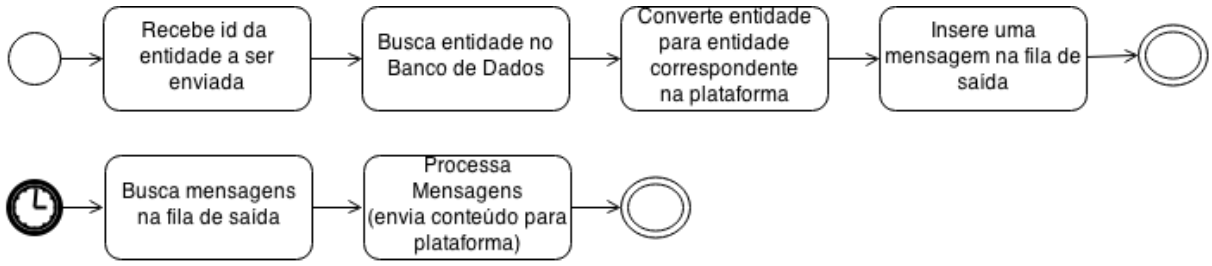

Figura 8 Fluxo de funcionamento do módulo de integração (envio de dados)

Basicamente, o módulo recebe um pedido de envio de um dado indicado pelo identificador único do dado no banco de dados. A seguir o dado é recuperado do meio de armazenamento e convertido em uma entidade da plataforma alvo da integração. Esta conversão é implementada neste módulo, o que resulta em módulos diferentes de integração para cada implementação realizada de cada um dos sistemas. Por fim o módulo de integração insere na fila de saída do sistema uma mensagem a ser enviada para a plataforma alvo.

Eventualmente, uma tarefa agendada no sistema recuperará todas as mensagens pendentes de envio e enviará cada uma para o seu destino.

No fluxo inverso (recebimento de dados), o comportamento é da seguinte maneira:

Figura 9 Fluxo de funcionamento do módulo de integração (recuperação de dados)

Neste sentido, o sistema ativamente recupera os dados que estão pendentes nas plataformas de mobilidade, e os insere na forma de mensagens na fila de entrada do sistema, que eventualmente será processada por outra tarefa agendada. O processamento de dados vindos de dispositivos mobile atualiza o conteúdo do banco de dados, que imediatamente já

fica disponível para consulta.

A única variante nestes processos é a forma como as entidades são convertidas para as plataformas alvo e o que é feito com os dados recebidos destas plataformas, que é diferente para cada cliente. Dessa forma, o trabalho também inclui a generalização deste módulo, de forma que o mesmo se torne independente do sistema com o qual está interagindo. O objetivo com isto é que o módulo de integração seja rigorosamente o mesmo para todos os clientes e produtos, com exceção de um único conversor, já que as plataformas de mobilidade são as mesmas para todos.

4.2.3 Utilização das tecnologias

Durante o processo de criação do X VENDAS e do X SERVIÇOS, a emprese detectou que muitas das tecnologias empregadas nestes produtos estavam sendo subutilizadas. Neste sentido, ficou combinado com a empresa o estudo e exploração destas tecnologias, de forma a tirar mais proveito das funcionalidades disponibilizadas pelas mesmas, principalmente JSPX (para criação de páginas geradas dinamicamente) e jQuery (para simplificar os scripts em javascript das páginas geradas).

Com o estudo das tecnologias, foi detectado, por exemplo, que os formulários das páginas poderiam ser populadas automaticamente em vez de manualmente (atualmente o programador escreve linhas de código para popular cada um dos campos das telas de listagem, criação e edição com seus valores padrão), e que diversos comportamentos implementados em javascript poderiam ser generalizados na forma de plugins jQuery próprios da empresa, de acordo com as necessidades que aparecessem.

4.3 Construção da nova topologia – definições

Os principais objetivos deste trabalho são reúso, flexibilização e boas práticas de engenharia de software. Desta forma, foi pensada uma topologia que permite o reúso das funcionalidades básicas implementadas no sistema, sem que haja cópia de código. A topologia aprovada pelo projetista da X para o G4 WORK foi a mostrada na Figura 10 abaixo.

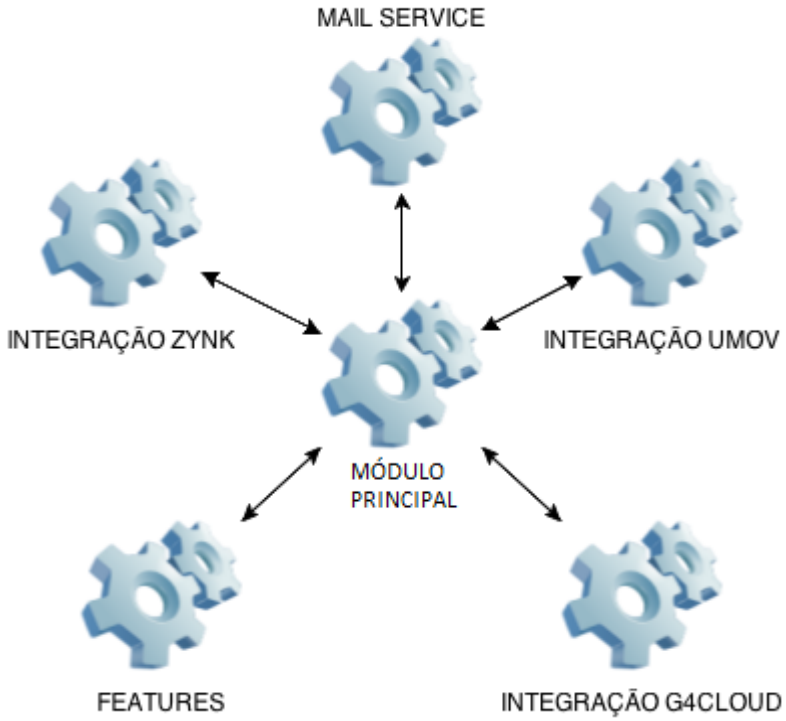

Figura 10 Topologia proposta e aprovada para o G4 WORK

Cada um dos sistemas representados na Figura 10 foi implementado na forma de uma aplicação web separada e independente das demais, permitindo assim o reúso integral de todos os módulos exceto o central. Cada módulo será implementado usando o padrão MVC, que é um padrão bem conhecido para o desenvolvimento de sistemas interativos (Leff, 2001). O MVC é uma instância da construção de aplicações web em 3 camadas mencionado no capítulo 2, exceto que nesta proposta as camadas não ficam presas a uma ordem hierárquica. Ele divide a aplicação nas camadas View (visão), Model (modelo) e Controller (controlador), sendo que a primeira é responsável pela visualização dos dados e, junto com a segunda, que processa os comandos do usuário, compõe a interface com o usuário; a terceira camada (Model) contém as informações exibidas pelas visões e a lógica que altera estas informações.

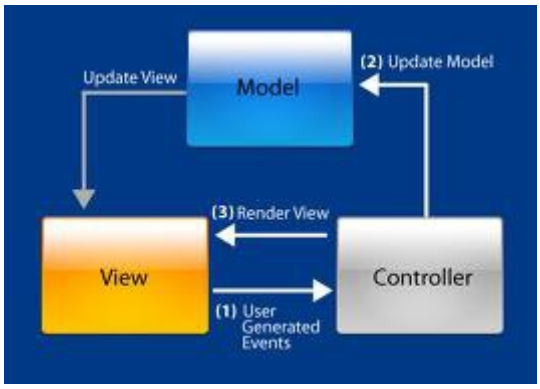

Figura 11: Relacionamento entre as camadas do padrão MVC

As próximas sessões mostram algumas definições feitas durante o projeto dos módulos e que ficaram acertadas com a empresa. Elas serão utilizadas durante o detalhamento dos módulos, e exemplificam as decisões tomadas para alcançar os principais objetivos deste trabalho: o reúso e a flexibilização.

4.3.1 Campos customizáveis

Para possibilitar a utilização dos componentes desenvolvidos no G4 WORK (em especial do módulo principal) para todos os produtos e clientes, foram adicionados a cada conceito diversos campos customizáveis. Estes campos possibilitam que sejam armazenadas informações específicas para cada implementação sem interferir com as outras implementações, mantendo-as o mais parecidas possível.

Foram inseridos campos de três tipos, que são descritos a seguir: um texto simples pode armazenar qualquer valor, como datas, booleanos, numerais etc, desde que convertidos para sua forma textual; um texto longo pode armazenar textos com tamanho indefinido, como descrições, observações etc; um campo Domínio armazena um valor pertencente a um domínio específico de cada produto. O tipo Domínio é melhor detalhado na sessão 4.3.4.

4.3.2 Parametrização

A fim de deixar o sistema mais flexível e dinâmico, foi projetado um modo de parametrização das funcionalidades para que as customizações mais comuns possam ser realizadas mais facilmente. Os parâmetros fazem variar o comportamento do sistema dependendo dos seus valores, de forma que não seja necessário desenvolver, testar e recompilar estes comportamentos variantes para cada cliente.

Esta funcionalidade é implementada através de uma tabela no banco de dados que guarda os valores dos parâmetros. Estes valores são, então, lidos em pontos chaves do programa, para tomar decisões sobre como proceder.

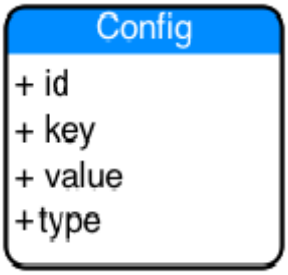

Figura 12: Representação Entidade-Relacionamento de um parâmetro do sistema

Alguns parâmetros do sistema podem ser vistos na tabela a seguir:

Chave	Valor	Tipo
system.use.umov	Y ou N	SYSTEM_PARAM
client.manage.address	Y ou N	SYSTEM_PARAM
user.list	{A, C, B, E}	PERMISSION

Tabela 1: Exemplos de parâmetros usados na flexibilização dos sistemas

Na Tabela 1 o primeiro parâmetro representa a necessidade ou não de o sistema enviar dados para a plataforma de mobilidade uMov.me. Para cada plataforma há um parâmetro indicando se o cliente para quem está sendo implementado o sistema vai utilizar os serviços da plataforma em questão. Os valores possíveis são Y e N, para sim e não (em inglês yes e no), respectivamente.

O segundo parâmetro indica se a entidade Cliente do sistema terá seus endereços gerenciados através do sistema ou se eles serão importados/integrados. Este parâmetro indica se, por exemplo, na tela de dados do cliente haverá uma opção de editar os endereços do cliente.

O terceiro parâmetro é um exemplo do controle de acesso ao sistema. Ele indica quais perfis de usuário terão acesso a quais sessões. Em específico ele trata da tela de listagem de usuário, onde podem ser vistos todos os usuários cadastrados no sistema. O valor do parâmetro é uma lista separada por vírgula das iniciais dos perfis que tem acesso à sessão. No caso A para Administrador, C para Coordenador, B para Back-Office e E para Executor.

4.3.3 Formas de comunicação

Como ficou definido que os módulos adjacentes ao sistema principal não teriam conhecimento das entidades implementadas pelo mesmo, foi necessário desenvolver um sistema de comunicação neutro que permitisse a troca das informações que cada um necessita.

O modelo de dados genérico utilizado é bastante simples, possibilitando seu emprego na comunicação entre qualquer par de módulos, sendo cada módulo responsável por traduzir os dados para o que estiver esperando.

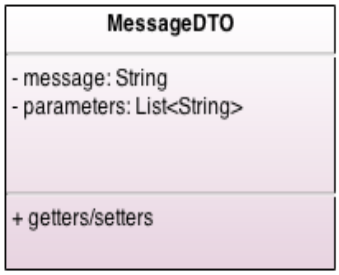

Figura 13: Modelo de dados utilizado na comunicação entre módulos

Esta estrutura é chamada um DTO (Data Transfer Object) pois serve justamente para transferir dados entre subsistemas de software. Ela carrega uma lista de informações ordenada, de forma que os conversores saibam em que ordem adicionar/ler as informações de que necessitam. O campo mensagem (“message”) carrega um texto qualquer, podendo ser um XML, um texto normal, um JSON ou outros, que é muito grande para ser um parâmetro.

No entanto, na comunicação entre dois módulos onde um é um módulo de integração

com uma plataforma móvel, apenas esta estrutura não é o suficiente. Logo, uma estrutura auxiliar para carregar os dados de uma entidade implementada no módulo principal para o módulo de integração, ou o inverso, foi desenvolvida. Ela é como segue:

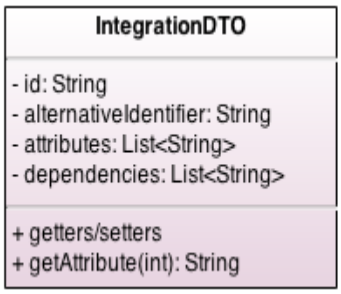

Figura 14: Estrutura auxiliar para carregar dados entre o módulo principal e os módulos de integração

Nesta estrutura, o campo id representa o identificador da entidade na plataforma alvo, enquanto o campo identificador alternativo representa o identificador da entidade no sistema. A lista de atributos contém as informações da entidade que serão enviadas para a plataforma, e a lista de dependências contém códigos de entidades dos quais esta depende (precisam existir na plataforma para que esta entidade também exista).

Estes dois modelos em conjunto possibilita, por exemplo, que sejam enviados para a plataforma de integração, diversas instâncias do mesmo conceito ao mesmo tempo. Isto facilita quando o envio destas instâncias é feito em lote para a plataforma de mobilidade.

4.3.4 Domínios

Em conjunto com a equipe da X, foi decidido manter a estrutura atual de “Domínios”. Esta estrutura possibilita que diversos pequenos domínios de dados sejam armazenados no banco de dados sem que haja necessidade de criar uma tabela para cada um. Esta estrutura utiliza uma única tabela com atributos comuns de forma que os domínios sejam diferenciados por um discriminador determinado pelo tipo do domínio.

A Figura 15 mostra a composição desta estrutura exibindo as características (atributos) dos Tipos de Domínio e dos Domínios. Logo abaixo, a Tabela 2 mostra alguns exemplos de possíveis dados representados nesta estrutura.

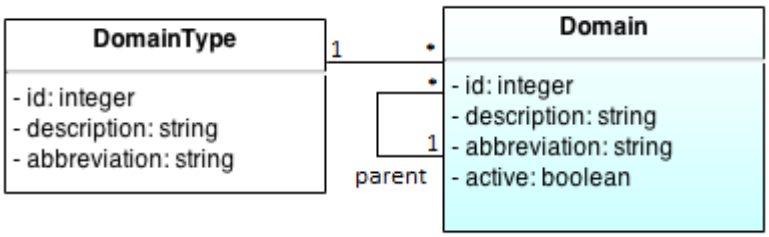

Figura 15: A estrutura representante de todos os domínios usados no sistema

Tipo de Domínio	Abreviatura
Estado	EST
Cidade	CID
Perfil de Acesso	PERF
Especialidade	ESPEC

Domínio	Abreviatura	Tipo	Domínio Pai
Rio Grande do Sul	RS	EST	
Porto Alegre	POA	CID	RS
Canoas	CAN	CID	RS
Administrador	ADM	PERF	
Coordenador	COORD	PERF	
Mecânico	MEC	ESPEC	
Eletricista	ELET	ESPEC	

Tabela 2: Exemplo de possíveis Domínios cadastrados no sistema

4.3.5 Filas

O sistema como um todo conta com uma fila de entrada, localizada no módulo principal, e n filas de saída, tantas quantos módulos de integração existentes. Os demais módulos não possuem filas, realizando os processamentos solicitados imediatamente à requisição.

O objetivo da implementação de filas é evitar a sobrecarga dos serviços que recebem requisições HTTP, transferindo o processamento pesado das informações para um processo paralelo que execute apenas esta tarefa. Este processo paralelo é registrado como uma tarefa agendada, para ser executado infinitas vezes com intervalo fixo de tempo entre elas.

Ambas filas apresentam comportamentos semelhantes, incluindo processamento, reprocessamento, e rejeição de mensagens, por exemplo. Sendo assim, foi estabelecida uma entidade genérica “Message”, que implementa estes comportamentos comuns. Esta entidade pode possuir filhas implementando comportamentos específicos, mecanismo conhecido como

Herança. Este relacionamento está ilustrado na Figura 16.

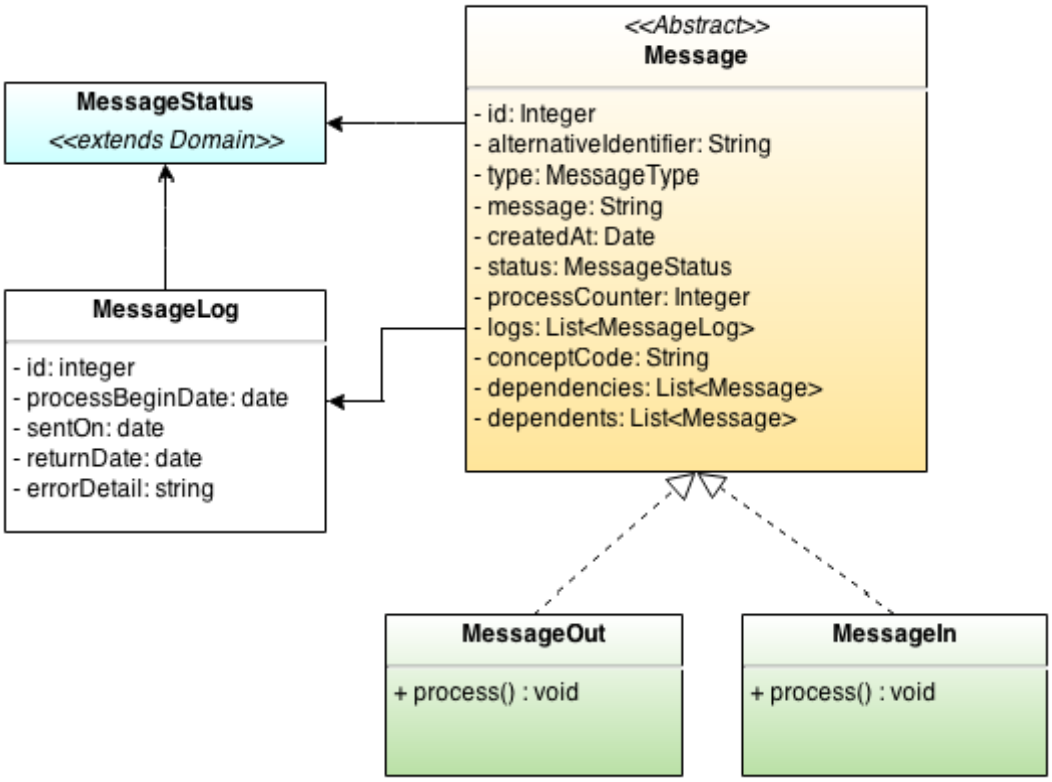

Figura 16: Generalização de Mensagens para evitar duplicata de código e atributos

A classe genérica (ou superclasse) “Message” possui todos os atributos que uma mensagem possui. Estes atributos incluem um contador de vezes que a mensagem foi reprocessada (em caso de erro), uma lista de registros (logs) de processamento e um código de conceito, que indica a qual entidade do módulo principal ela se refere.

Em relação ao modo atual de filas utilizado pela empresa (entidades “MessageIn” e “MessageOut” independentes), este modelo evita a duplicidade de código que implementa os comportamentos semelhantes já exemplificados e facilita a inserção/remoção de atributos e comportamentos das mensagens. Também foi criado o conceito de dependência entre mensagens, com a inserção das listas de mensagens dependentes (que não podem ser processadas antes desta) e de mensagens das quais esta depende (que devem ter sido processadas antes desta). Esta necessidade foi reportada pela equipe de suporte da empresa, quando percebeu que várias mensagens ficavam com erro e eram reprocessadas inúmeras vezes em decorrência de uma única mensagem com erro.

4.3.6 Integração com plataformas de mobilidade

Devido ao alto grau de semelhança entre os módulos de integração, foi especificado um modelo genérico da composição destes módulos, que segue abaixo.

Figura 17: Composição genérica de um módulo de integração

Pode-se ver na Figura 17 que cada módulo possui: um controlador próprio, responsável por receber requisições externas; um agendador de tarefas para controlar o processamento da fila de saída; um conversor da estrutura auxiliar "IntegrationDTO" para uma entidade da plataforma; uma fila de saída, com telas de visualização das mensagens na fila, possibilitando processamento manual, verificação de logs etc; a modelagem das entidades da plataforma alvo.

4.3.7 Validação de formulários

O processo de validação dos dados preenchidos em formulários em diversas partes do sistema – grande fonte de divergência entre os sistemas – foi repensada e padronizada. Ela é compreendida de três partes, sendo a primeira feita no lado cliente da aplicação, via código javascript, e as outras duas no lado servidor.

A primeira parte da validação dos dados do formulário consiste em validar superficialmente o preenchimento dos dados. Exemplos destas validações superficiais incluem, por exemplo, se todos os campos de preenchimento obrigatório foram preenchidos,

se uma data está no formato correto, se duas datas estão em ordem cronológica etc. São validações realizadas sem nenhum conhecimento sobre os dados previamente armazenados pela aplicação, e que portanto dispensam a necessidade de uma consulta ao servidor da aplicação e ao banco de dados.

A segunda parte é a validação de erros nos dados em relação aos dados armazenados no banco de dados previamente. Esta validação ocorre no servidor, pois apenas este tem acesso à interface de comunicação com o banco de dados. Ela valida inconsistências que não permitem o salvamento dos dados, como por exemplo a unicidade de um campo, conflitos entre dados etc. O usuário só pode concluir a ação após alterar os valores problemáticos para outros que não disparem erros nesta parte da validação.

A terceira e última parte valida situações em que o usuário deve ser alertado, mas pode prosseguir se assim desejar. Ela também precisa de contato com o banco de dados, pois valida principalmente regras de negócio e ações executadas em tempo real, por exemplo, se o estado de uma entidade mudou enquanto o usuário a estava editando.

Caso os dados do formulário passem em todas as etapas da validação, são automaticamente salvos, sem a necessidade de nova chamada ao controlador. Este procedimento diminui o retardo no processo das informações já que, no melhor caso, realiza apenas uma chamada HTTP Rest para o servidor.

O fluxo das validações descrito acima está ilustrado na Figura 18.

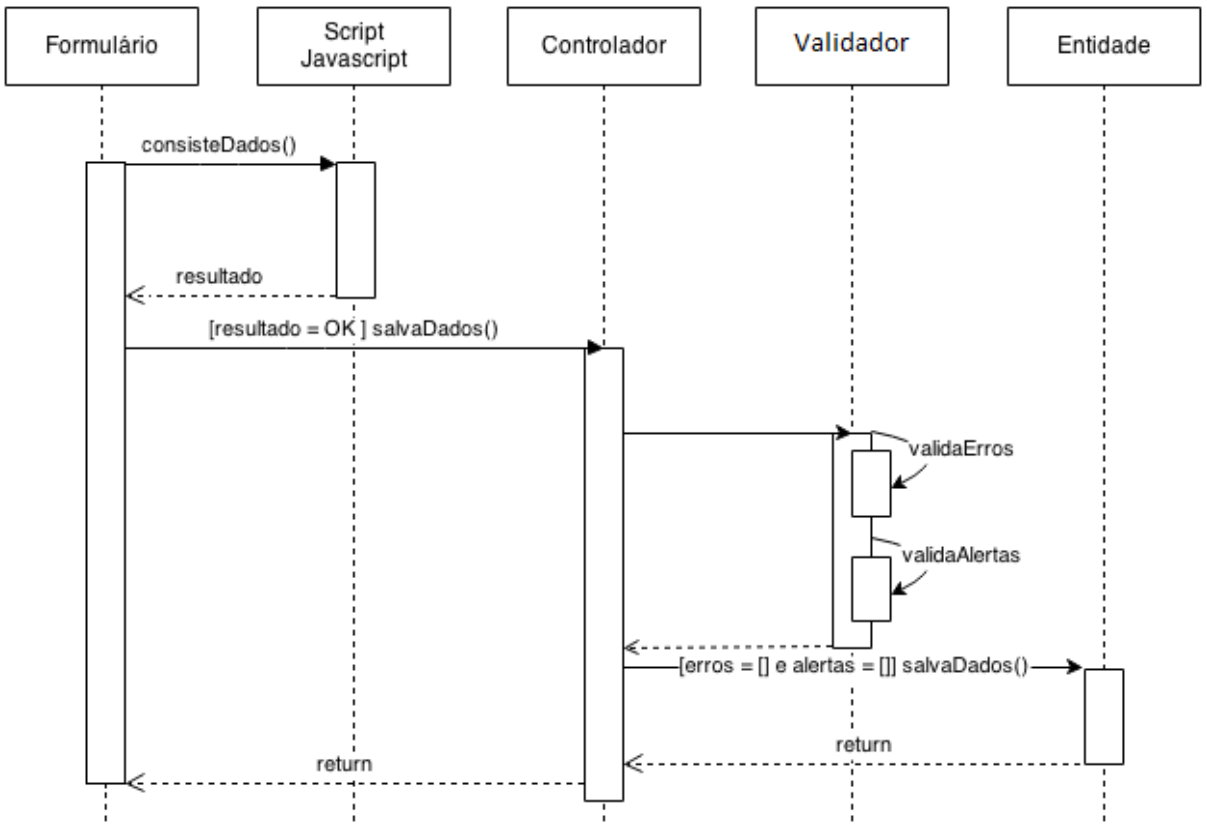

Figura 18: Diagrama de sequência do fluxo de validação padrão implementado no G4 WORK

4.3.7 Plugins jQuery

Outra fonte de problemas nas implementações VENDAS e SERVIÇOS foi o alto grau de duplicação de código em javascript, que é de difícil manutenção devido às poucas opções de bons depuradores. Funcionalidades para realizar procedimentos comuns, como abrir uma popup(modal), verificar se todos os campos foram preenchidos em um formulário, exibir um mapa do Google Maps e outras foram replicadas a cada necessidade, ocasionando difíceis alterações em comportamentos comuns.

O javascript é utilizado, nas soluções X, em conjunto com o jQuery, uma biblioteca leve "escreva menos, faça mais" cujo propósito é tornar mais fácil a utilização de javascript nas páginas da web (W3C, 2014). O jQuery possibilita a criação de plugins (componentes de software que podem ser acoplados a sistemas existentes para agregar funcionalidades) que podem ser utilizados ao longo de todas as páginas do sistema, generalizando os comportamentos de estruturas de tela. Inicialmente foi definida a criação de 4 plugins:

- Plugin de modais – para generalizar a criação de modais em todo o sistema dependendo do seu tipo: mensagem, pergunta, exibição ou formulário.
- Plugin de mapas – para generalizar a exibição de mapas do Google Maps, a plotagem(exibição de marcadores) de endereços, a roteirização de pontos etc
- Plugin de tabelas – para generalizar a criação de tabelas dinâmicas com paginação e classificação
- Plugin de componentes diversos – para generalizar a criação de componentes como listas de seleção múltiplas, listas reordenáveis e campos com contagem de caracteres

4.3.8 Flexibilização de formulários

Hoje os formulários de qualquer tela são construídos manualmente por um programador da X com base nos campos do cadastro da entidade que será editada/criada/listada. Cada formulário possui um DTO correspondente para transportar os dados do formulário para o controlador responsável por executar a ação desejada sobre os dados. Desta forma, para cada campo alterado, adicionado ou removido de um formulário, o DTO que carrega os valores dos campos tem que ser atualizado manualmente. Os dados do DTO recebidos pelo controlador são passados para a entidade responsável que então tem que transferí-los para a entidade que será salva (no caso de uma criação ou atualização) ou usá-los na construção de um SQL dinâmico para a consulta ao banco de dados. Tanto a transferência para a entidade a ser salva quando a construção do SQL dinâmico também tem que ser alteradas manualmente.

Um dos desejos da empresa para melhoria dos produtos é ter estas etapas configuráveis através do banco de dados, de forma que seja possível inserir e remover campos dos formulários sem necessidade de atualizar o código diretamente. Sendo um dos objetivos deste trabalho a flexibilização, foi pensada uma estrutura inicial para ser um primeiro passo na direção da realização deste desejo.

No primeiro momento foi definido como objetivo a criação dinâmica dos campos das listagens. Desta forma, novos campos pré-definidos podem ser incluídos dinamicamente

através do banco de dados e do arquivo de rótulos do sistema (um arquivo tipo chave/valor que possui os valores dos rótulos utilizados no sistema para possibilitar a internacionalização). Também foi alterado o sistema de filtros para despoluir a tela a desejo da empresa. Ao contrário da abordagem atual, onde todos os filtros são exibidos na tela, a nova abordagem permite que sejam exibidos alguns filtros, e que o usuário, se assim desejar, possa adicionar outros filtros.Uma comparação entre o sistema de filtros antigo e o novo pode ser vista nas Figuras 19 e 20, e a modelagem dos filtros pode ser vista na Figura 21.

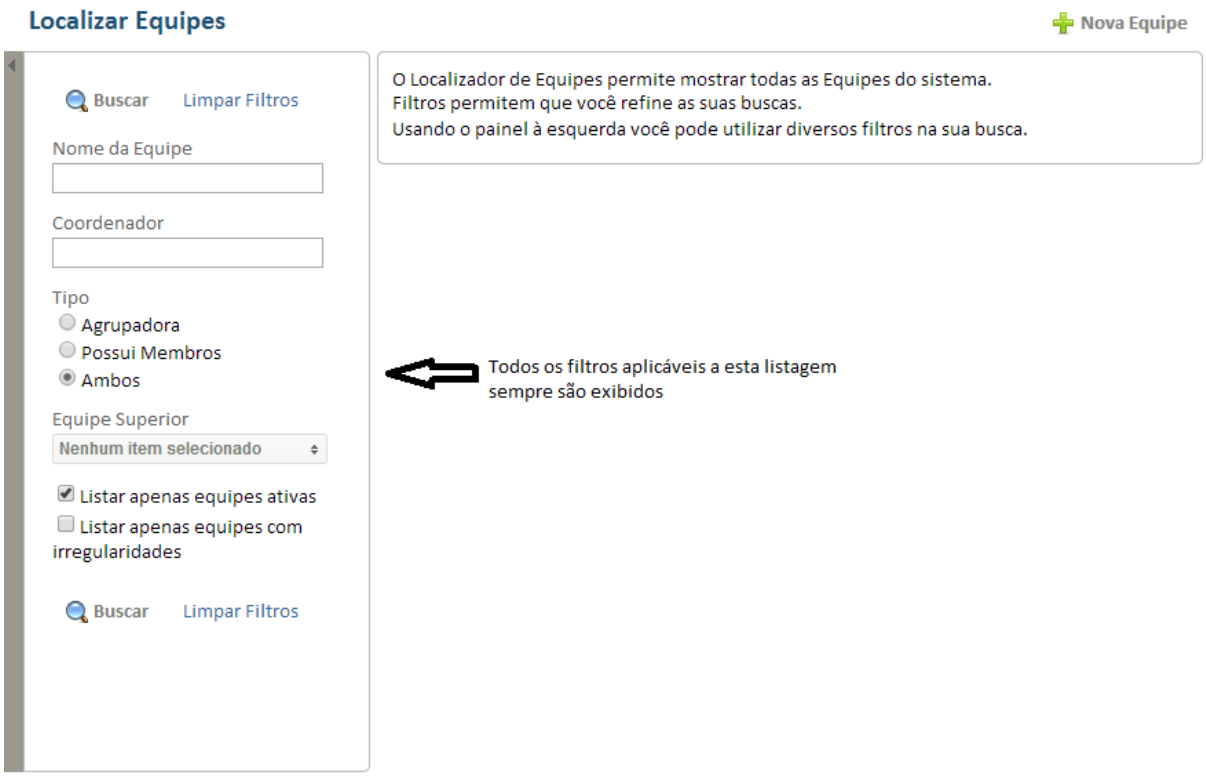

Figura 19: Antigo sistema de filtros dos produtos X

Figura 20: Novo sistema de filtros dos produtos X

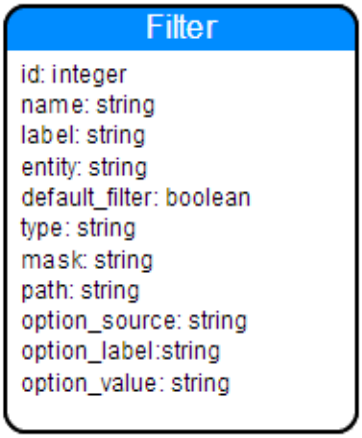

Figura 21: Modelo ER de armazenamento de filtros dinâmicos

Nesta modelagem, os campos nome e rótulo se referem a chaves no arquivo de rótulos. O campo nome é o texto exibido quando o usuário visualiza os filtros que podem ser adicionados à listagem caso o filtro não seja padrão, e o campo rótulo é o rótulo do filtro quando está sendo exibido.

O campo entidade define a qual listagem o filtro se aplica, por exemplo “user”, “team”, “client” etc. O tipo do filtro define como ele será exibido na tela e recebe um nome de uma estrutura HTML, por exemplo “TEXT”, “SELECT-ONE”, “RADIO”, “CHECKBOX”, etc.

A máscara se aplica apenas se o tipo do filtro for “TEXT”, e serve para que o campo possua uma máscara de formato.

O caminho é o nome da variável, no DTO correspondente à filtragem desta entidade, que vai receber os valores do filtro em questão;

A fonte de opções se aplica a filtros do tipo “SELECT-ONE”, “RADIO”, “SELECT-MULTIPLE”, e indica o nome da variável JSPX que contém os valores a serem exibidos para este filtro. Os campos rótulo da opção e valor da opção são os nomes dos atributos que serão utilizados respectivamente como rótulo e o valor das opções exibidas.

Utilizando esta abordagem do modo como está, ainda é necessário editar o DTO e a criação do SQL de busca manualmente. No entanto, a criação do código HTML da página é dinâmica, a ativação/desativação de campos pode ser feita via banco de dados e a troca de rótulos dos filtros via arquivo de rótulos. Nenhum destes passos necessita mais alteração direta de código.

A Tabela 3 mostra a configuração dos filtros da tela de equipes, vista na Figura 20. Em seguida é mostrada a modelagem do DTO que carrega os valores dos filtros (Figura 22) e um trecho de código do controlador que popula as opções dos filtros para análise.

Nome	Rótulo	Entidade	Filtro Padrão	Tipo	Máscara	Caminho	Fonte de Opções	Valor da opção	Rótulo da opção
	label.teams.name	team	true	TEXT		name			
	label.teams.coordinator	team	true	TEXT		coordinator			
	label.teams.type	team	true	RADIO		type	typesList	custom1	description
	label.teams.parent	team	false	SELECT-MULTIPLE		parents	parentsList	id	name
label.teams.active	label.teams.onlyActive	team	false	CHECKBOX		onlyActive			
label.teams.irregular	label.teams.onlyIrregular	team	false	CHECKBOX		onlyIrregular			

Tabela 3: Configuração via Banco de dados dos filtros de Equipe

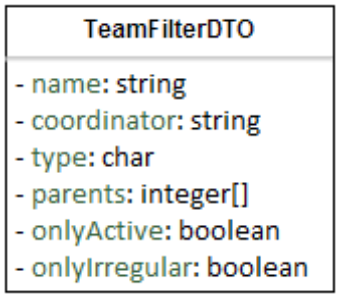

Figura 22: Representação do DTO que carrega os dados do filtro de Equipes para o controlador de Equipes

```
TeamController.java
(...)
map.put("parentsList", Team.findTeamsByType(TeamType.NODE));          //busca no banco de dados todas as Equipes que são do
                                                                      //tipo NODO, ou seja, são parte de uma hierarquia de
                                                                      //Equipes, mas não são folhas e disponibiliza para a tela
                                                                      //uma variável com o resultado
map.put("typesList", Domain.findDomainsByType(TeamType.TYPE_ABRV));   //busca no banco de dados os tipos possíveis de Equipes
                                                                      //e disponibiliza para a tela uma variável com o resultado
(...)
```

5 DESENVOLVIMENTO

Esta sessão detalha partes do desenvolvimento das aplicações web que compõe o G4 WORK, particularidades sobre a implementação e comentários sobre a necessidade das alterações.

5.1 Tecnologias

Serão utilizada para desenvolvimento as tecnologias já adotadas na empresa para as soluções. São elas:

- Banco de dados MySQL, opção open-source de gerenciador banco de dados, o segundo mais utilizado no mundo (SOLID IT, 2014).
- Hibernate, por ser um framework de abstração e persistência de dados que mascara o acesso ao Banco de Dados, tornando a aplicação independente do mesmo.
- Linguagem de programação Java, por ser multiplataforma e oferecer uma grande diversidade de componentes, repositórios e frameworks, bem como a possibilidade de desenvolver sistemas voltados para web.
- Framework Spring, por ser um framework Java amplamente utilizado, disposto em módulos separados voltados para tipos de aplicação específicos.
- A ferramenta de produtividade Spring Roo, para agilizar a criação dos conceitos básicos do sistema assim como a camada DAO dos mesmos.
- A IDE STS, que já vem com suporte a todas as tecnologias necessárias, facilitando a configuração e parametrização das mesmas.
- A tecnologia JSPX que possibilita a geração dinâmica de páginas da web.
- As ferramentas HTML e CSS para formatação das páginas da web geradas pelo sistema.
- A linguagem de programação client-side javascript, por possibilitar a criação

de páginas da web dinâmicas e mais amigáveis

- jQuery, a biblioteca javascript mais usada no mundo (W3TECHS, 2014), para facilitar a inclusão de comportamentos nas estruturas HTML das páginas geradas pelo sistema
- O conjunto de tecnologias agrupadas sob o nome Ajax (GARRETT, 2005), para possibilitar a interação assíncrona do cliente com o servidor.

A Figura 23 mostra a interação entre as tecnologias listadas acima.

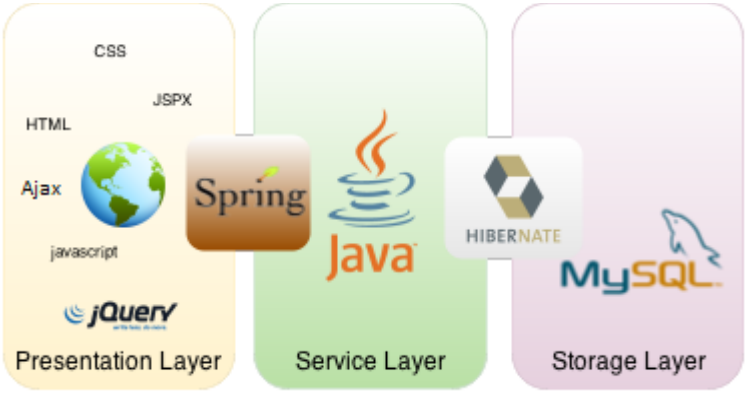

Figura 23: Tecnologias utilizadas no G4 WORK

5.2 Funcionalidades Visadas

Aqui serão descritas brevemente as funcionalidades que foram incluídas dentre os objetivos de desenvolvimento neste trabalho, bem como detalhes da sua implementação relacionados a decisões de projeto e estágio da implementação. Não é objetivo deste trabalho implementar na íntegra todas as regras de negócio estabelecidas para as soluções, mas sim aquelas funcionalidades que são comuns a todos os produtos e que servirão de base para implementação de novas regras e funcionalidades dentro da nova organização. Sendo assim, foi estabelecida como meta a implementação efetiva de algumas partes dos módulos assinalados na Figura 24.

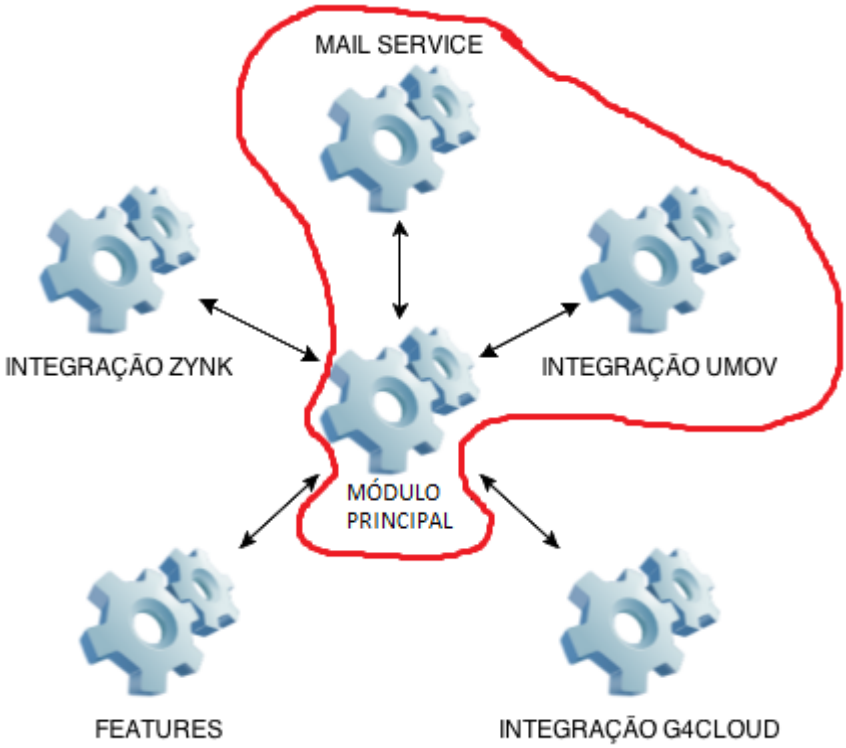

Figura 24: Módulos objetos da parte de implementação deste trabalho

Estes módulos foram escolhidos por oferecerem a maior quantidade de implementações diferentes, incluindo: a implementação de plugins no módulo principal, assim como manutenção de entidades; um exemplo de integração com uma plataforma de mobilidade, no caso o uMov.me, cuja implementação será bastante semelhante para os outros módulos; a implementação do envio de e-mails gerados pelo sistema de forma dinâmica e independente, para que seja possível editar o conteúdo dos mesmos facilmente sem necessidade de recompilação (utilização de arquivos de conteúdo).

O módulo Features servirá para oferecer à aplicação web funcionalidades padrão a todas as soluções e que podem ser tratadas individualmente, como o upload/download de documentos, e a manutenção de checklists (listas de perguntas enviadas ao dispositivo móvel para complementar as informações inseridas pelo usuário Executor). No entanto ele não foi listado dentre as implementações propostas neste trabalho pois os conhecimentos e padrões estipulados para os módulos listados valem igualmente para este.

5.2.1 Módulo Principal

A seguir são melhor descritas as funcionalidades implementadas para cada um dos três módulos, comparando com as implementações das mesmas nos outros produtos já em comercialização (X VENDAS e X SERVIÇOS).

5.2.1.1 Gerência de Usuário

A gerência de Usuários compreende a criação, edição, e listagem de usuários. Entretanto, o usuário de perfil Executor possui diversas características a mais que os outros perfis, relacionadas às suas atividades. Apesar de ser um usuário da solução X, os usuários do tipo Executor podem ou não ter acesso à aplicação web. No caso do X SERVIÇOS e do X LOGÍSTICA, os usuários deste perfil possuem área de manutenção diferenciada dos outros usuários, pois são papéis integralmente externos (nunca estão na sede da empresa). Em contraponto, no X VENDAS e no X AGRO os usuários Executores podem desempenhar atividades na própria sede da empresa, justificando seu acesso à aplicação web e à visualização dos dados contidos na mesma. Neste último caso o Executor visualiza as mesmas informações que os outros usuários, tendo impostas apenas algumas restrições de acesso.

O G4 WORK foi pensado como um núcleo de funcionalidades comuns a todos os produtos da X e em alguns produtos o usuário de perfil Executor pode vir a ter acesso à aplicação web. É possível dizer que os usuários Administradores, Coordenadores e Back-Offices compreendem a maioria dos usuários do módulo principal mas não necessariamente todos. Desta forma, a modelagem de usuários precisa permitir estas diferenças.

Nos produtos já desenvolvidos (X VENDAS e X SERVIÇOS), a modelagem de usuários é diferente para cada um. Enquanto no X VENDAS os usuários são modelados como uma única entidade, com propriedades referentes ao acesso a dispositivos móveis e à aplicação web juntos, no X SERVIÇOS os usuários Executores foram separados completamente dos usuários comuns. Ambas abordagens trazem problemas: a primeira traz o excesso de propriedades não utilizadas, já que a mesma estrutura contém atributos referentes tanto ao acesso à aplicação web (para usuários Administradores, Coordenadores, e Back-Offices) quanto ao acesso aos aplicativos móveis (para usuários Executores); e a segunda traz

dificuldades pois não permite facilmente que um usuário possua tanto acesso aos aplicativos móveis quanto à aplicação web.

O G4 WORK propõe, então, uma nova modelagem, que permite a um usuário possuir simultaneamente acesso às plataformas móveis e ao sistema web, sem repetição de atributos e de forma independente dos demais usuários. Por exemplo, um usuário Administrador pode possuir acesso aos aplicativos móveis e outro não, e o mesmo para um usuário Executor qualquer. Esta nova modelagem será apresentada a seguir.

5.2.1.1.1 Gerência de Usuários Administradores, Coordenadores e Back-Offices

Estes perfis são responsáveis por gerenciar os dados armazenados na aplicação web, efetuando atualizações nos cadastros e nas atividades mantidas no sistema. Estes usuários permanecem sempre na sede da empresa contratante do produto, e não realizam tarefas nos dispositivos móveis. É possível que eles tenham acesso ao conjunto de aplicativos X Apps, para visualização de dados apenas.

A nova modelagem de dados proposta neste trabalho para um usuário com um destes perfis é exibida na figura abaixo:

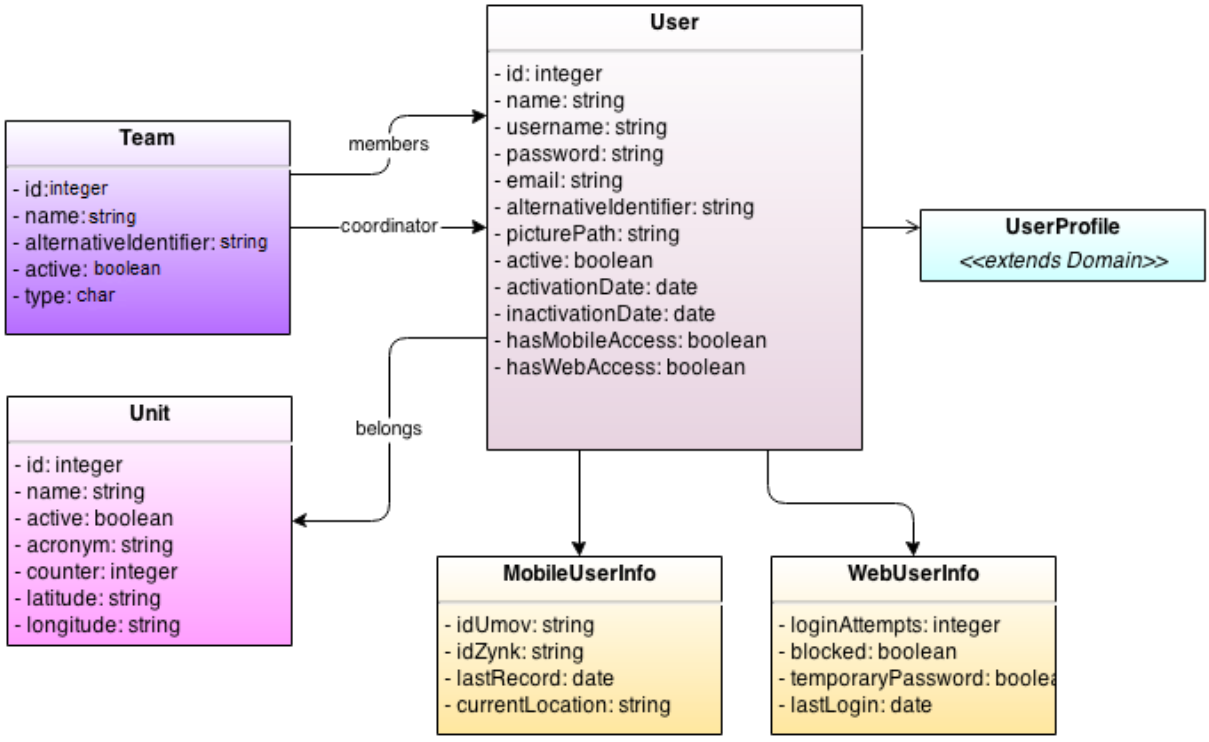

Figura 25: Representação dos dados de um usuário não Executor

Na Figura 25 pode-se observar que os atributos para acesso à aplicação web e aos aplicativos móveis ficam separados do usuário (WebUserInfo e MobileUserInfo). Desta forma, um usuário pode possuir qualquer combinação destes dois acessos independentemente do seu perfil (UserProfile). Um usuário também fica associado a uma Unidade da sua empresa e a uma equipe, que podem ser utilizadas de forma independente para limitar o acesso do usuário aos dados do sistema.

5.2.1.1.2 Gerência de Usuários Executores

Os usuários executores são aqueles que levam consigo um dispositivo móvel a fim de executar nele alguma tarefa para a empresa onde está implantada a solução X. Eles possuem todas as características de um usuário dos outros perfis, mas possuem diversas particularidades devido à natureza diversa das suas atividades.

A fim de proporcionar para a empresa contratante maior controle sobre a mobilidade dos seus funcionários, é necessário armazenar, por exemplo, a atividade atual de cada um e a sua localização. Para este perfil também é armazenado um endereço, caso a empresa permita deslocamentos partindo ou terminando em um endereço do funcionário, por exemplo, sua casa.

Outras particularidades incluem a disponibilidade do usuário (disponível, em atendimento, indisponível para atender, etc), sua especialidade (que tipos de serviço ele pode atender), e o seu local de trabalho. Na maior parte dos casos, o usuário executor trabalha fora da empresa contratante do produto X. Ele se desloca até um cliente, realiza o serviço e retorna para a empresa. Entretanto, em alguns casos, usuários executores podem realizar seus serviços na própria empresa, por exemplo, em uma oficina. Neste caso seu local de trabalho não é mais “campo” e sim “interno”.

Abaixo, a Figura 26 representa a nova modelagem do relacionamento entre as informações armazenadas para um usuário executor.

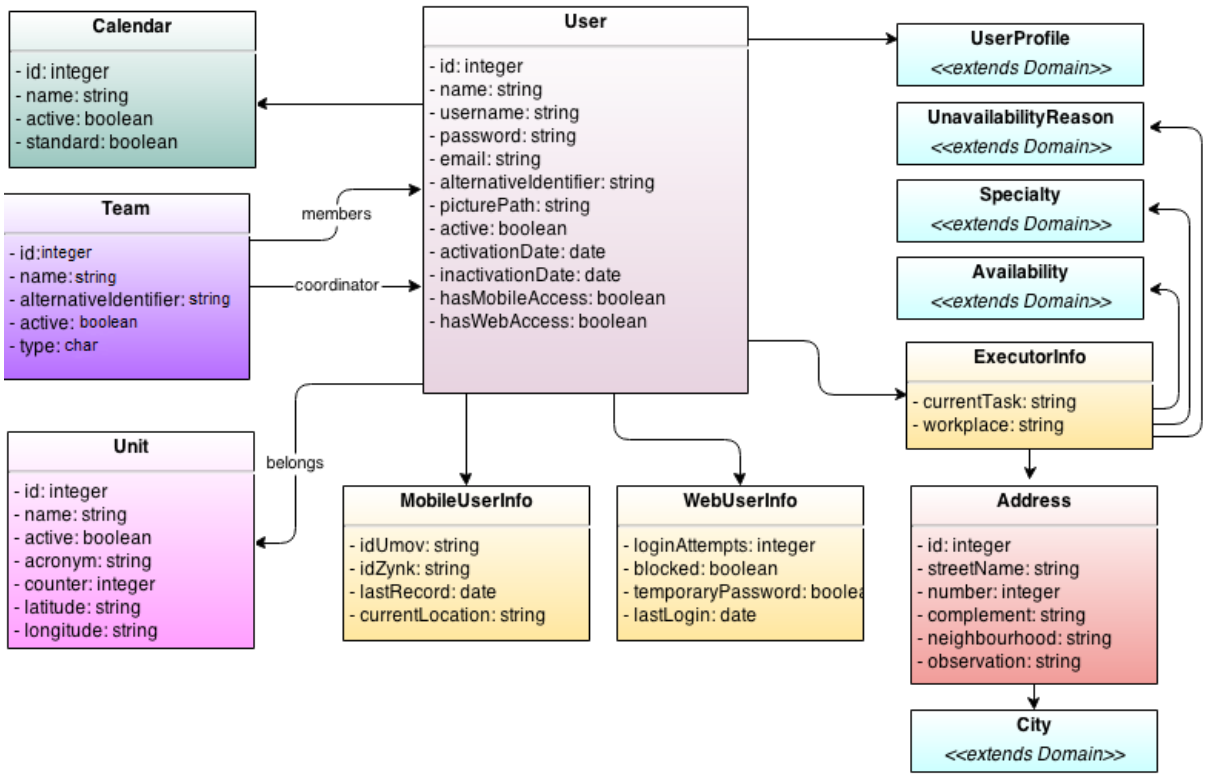

Figura 26: Representação dos dados de um usuário Executor

Como é possível identificar na Figura 26, para usuários com perfil Executor é acrescentada uma estrutura extra, apenas com informações referentes ao trabalho do mesmo (ExecutorInfo), e um Calendário.

5.2.1.1.2.1 Gerência de Calendários

Usuários com perfil Executor possuem mais uma característica distinta dos outros usuários, que é o cadastro dos seus calendários de trabalho. O calendário representa os dias e horários, em um determinado período, em que o usuário estará à disposição da empresa, de forma que apenas aqueles executores disponíveis sejam cogitados para realização de determinado serviço.

O calendário também pode ser utilizado para cálculo de horas extras a serem pagas ao funcionário, já que possibilitam a obtenção de registros sobre serviços realizados fora dos horários de trabalho previamente acordados. Assim, a empresa tem mais controle sobre seus gastos, podendo dar preferência aos funcionários em horário de trabalho no lugar daqueles em que teria de pagar hora extra.

Vários calendários podem ser cadastrados no sistema, mas apenas um será escolhido como padrão. Este será utilizado em cálculos onde nenhum calendário foi escolhido. Cada Executor pode possuir no máximo um calendário ativo por período de tempo.

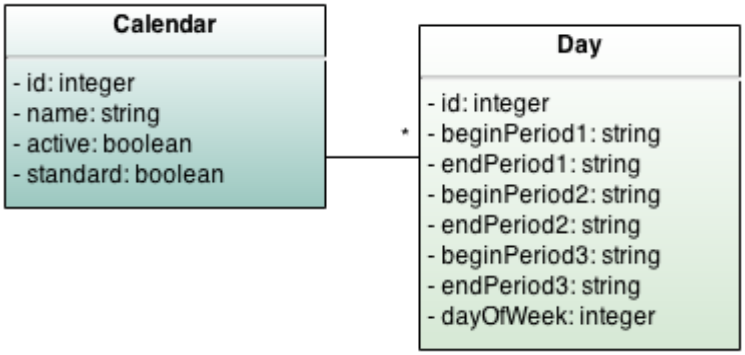

Figura 27: Representação dos dados de um calendário

O trabalho feito no G4 WORK para a gerência de calendários foi a reimplementação das funcionalidades incluindo a padronização das validações dos dados inseridos, a otimização da criação de código HTML da tela de criação/edição de calendário e a limpeza do modo de envio dos dados. Nas implementações atuais o envio dos dados do calendário é improvisado e ineficiente devido ao mau entendimento da ferramenta DataBinder do framework Spring, que é encarregada de transformar os dados enviados via HTTP Rest para o formato adequado para o consumo dos mesmos pelo controlador dos calendários.

5.2.1.2 Gerência de Equipes

As equipes servem como agrupadoras de usuários, sendo úteis para filtragem dos mesmos e conforme necessidade do cliente, como restritoras de acesso. Se a empresa contratante da solução assim desejar, restrições ao acesso dos dados de outras equipes podem ser ativadas. Com estas restrições, os usuários Administradores continuam tendo aceso a todos os dados, os Coordenadores tem acesso às suas equipes e a toda a árvore hierárquica de cada uma delas, e os Back-Offices só veem os dados referentes às suas equipes.

Existem dois tipos de equipe: as equipes Agrupadoras, que não possuem usuários associados, mas são superiores a outras equipes; e as equipes com membros, que possuem apenas usuários associados, e nenhuma equipe subordinada. Desta forma é possível

estabelecer uma hierarquia de acesso com quantos níveis forem necessários à empresa contratante.

Cada equipe, independente do seu tipo possui um usuário Coordenador associado a ela e, se for do tipo "com membros", um número indefinido de membros Back-offices e Executores. Usuários Administradores não são associados a equipes pois possuem acesso a todos os dados do sistema incondicionalmente. O Coordenador de uma equipe é quem gerencia a equipe, adiciona e remove membros e tem acesso a todos os dados da mesma de forma consolidada. Ele pode ser alterado por qualquer usuário Administrador, já que estes possuem controle sobre todo o sistema. A estrutura básica de uma equipe foi exibida nas Figuras 25 e 26.

Os membros da equipe são alterados através de um formulário de busca de candidatos a membro da equipe. Os usuários resultantes desta busca são exibidos em uma lista. Ao lado desta são exibidos os atuais membros da equipe. A alteração, então, ocorre através das opções "Adicionar" e "Remover", como pode ser visto na Figura 28 abaixo:

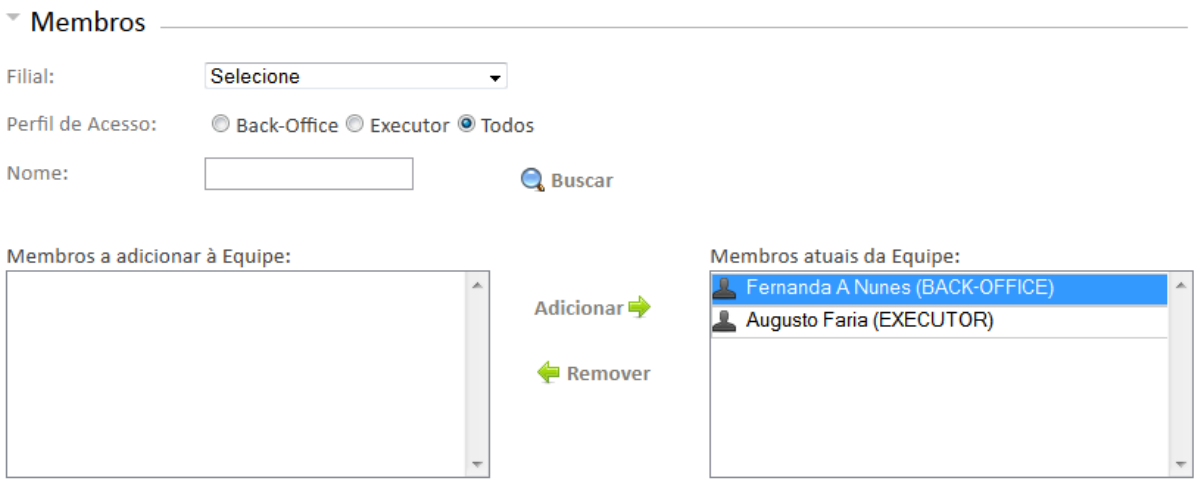

Figura 28: Edição dos membros de uma equipe

A implementação desta funcionalidade nos sistemas X VENDAS e X SERVIÇOS é funcional mas não satisfatória. Ela acontece de forma transacional, isto é, após alterar os membros da equipe em tela, o usuário efetuando estas alterações pode ou não escolher salvá-las. Este comportamento, no entanto, dá margem para diversos problemas, incluindo a perda de dados por força externa (queda de energia, travamento do navegador ou da máquina etc) e a ocorrência de usuários fantasma (um usuário que em tela é removido da equipe mas, como a

alteração não foi salva, não aparece mais como possível candidato a membro, impossibilitando que ele seja novamente adicionado à equipe).

Foi definido com a empresa que operações deste tipo serão feitas isoladamente, de forma que cada inserção/remoção faça efeito imediato no sistema. Esta abordagem soluciona os problemas relatados.

5.2.1.3 Gerência de Clientes

A gerência de clientes é a primeira parte do módulo principal que agrega conhecimento à empresa. A partir dele é possível cadastrar, editar e consultar clientes, armazenando informações importantes como endereços, contatos e telefones. Este é o primeiro passo para o aproveitamento integral das facilidades oferecidas pelos produtos que terão o G4 WORK como base.

A maior melhoria implementada para a gerência de clientes no G4 WORK foi a criação de controladores separados para cada conceito relacionado, mesmo os que só existem associados a um cliente, como o conceito Contato. Ao contrário do conceito Endereço, por exemplo, que pode estar associado a um Cliente ou a um Executor, o Contato só existe e é mantido através do cadastro de clientes. Por este motivo, nas soluções existentes da X o contato é administrado pelo mesmo controlador que o administra o cliente. A separação dos conceitos (criação de controladores separados) foi uma decisão tomada em função da possibilidade futura da inserção de contatos em outros conceitos do sistema. Hoje, as ações tomadas em cima de contatos poluem o controlador de clientes e, caso seja necessário passar a associar contatos com outros conceitos (como aconteceu com endereços), esta associação será difícil por causa do alto acoplamento com a entidade Cliente.

Como visto nos mapas de conceito (Figuras 4, 5 e 6) da sessão 4.2.1, os clientes estão associados a outros conceitos, como Conta (VENDAS), Ordem de Serviço (SERVIÇOS), Romaneio (LOGÍSTICA) e Produção (AGRO). No entanto, como estes conceitos são específicos de cada solução, eles não são abordados neste trabalho.

Os conceitos relacionados à entidade Cliente alvo deste trabalho podem ser vistos na Figura 29.

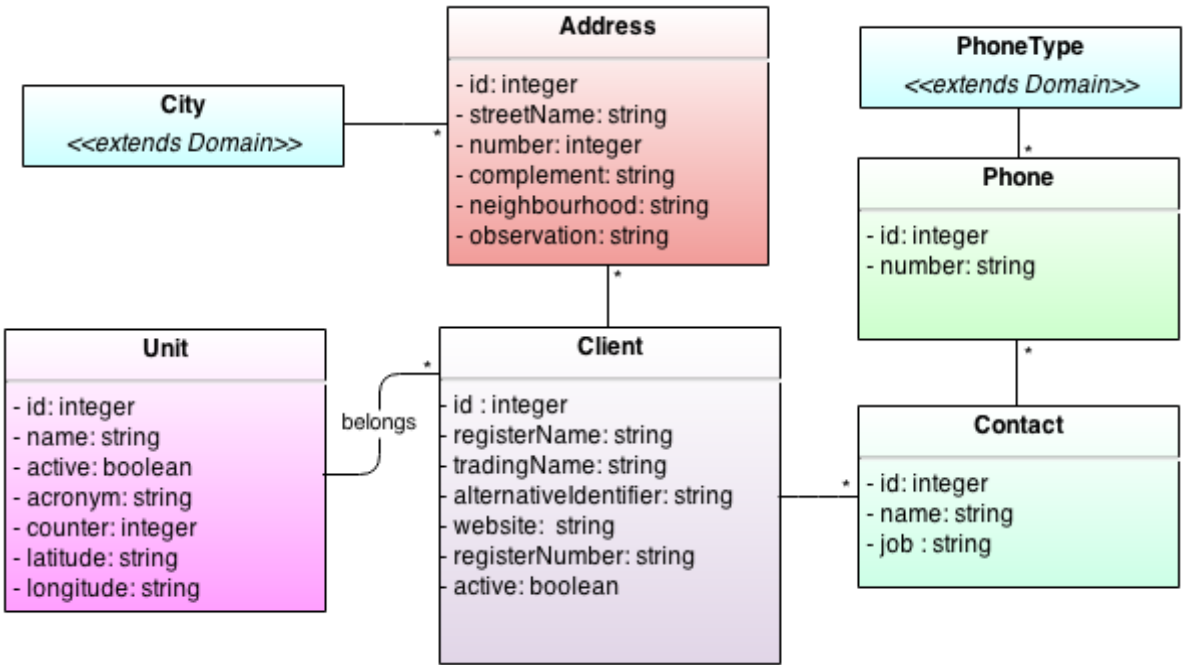

Figura 29: Representação dos dados de um Cliente

5.2.2 MailService

Este módulo é o responsável pelos envios de e-mail do sistema. Existem diversas situações onde e-mails são enviados pelo sistema para destinatários específicos. O mais importante deles é o e-mail de alerta, onde o sistema alerta a equipe de suporte da X sobre possíveis problemas nos fluxos de execução. Este disparo ocorre, por exemplo, quando uma mensagem na fila é processada um número fixo de vezes (que depende do seu tipo) sem sucesso. Nestes casos é necessária intervenção da equipe de suporte para identificar o problema e possivelmente solucioná-lo.

Outros e-mails são enviados naturalmente como parte do fluxo de execução, por exemplo na criação de um usuário. Todos os usuários são inicialmente criados com uma senha aleatória gerada automaticamente pelo sistema. Esta senha é enviada para o e-mail do usuário cadastrado, junto com instruções de como acessar o sistema web ou ao aplicativo móvel, dependendo das permissões de acesso configuradas para o usuário.

Há ainda um número finito de e-mails cujo conteúdo a empresa permite que o cliente, para quem a solução é implantada, defina. Todos os e-mails, tanto os personalizados quanto os fixos, são parametrizados através do arquivo de propriedades do módulo, de forma que não é

necessário recompilar o código para alterar seu conteúdo ou destinatário.

Um trecho do arquivo de propriedades deste módulo está exibido abaixo.

create.user.web.subject=Bem-vindo ao X <SOLUÇÃO>!
create.user.web.content=Olá, seja bem-vindo ao X <SOLUÇÃO> ! Seguem abaixo os seus dados e instruções para acessar a aplicação web. (...)
create.user.mobile.subject=Bem-vindo ao X <SOLUÇÃO>!
create.user.mobile.content=Olá, seja bem-vindo ao X <SOLUÇÃO> ! Seguem abaixo os seus dados e instruções para acessar a aplicação web. (…)
alert.unexpectedError.subject= Erro inesperado no X <SOLUÇÃO>
alert.unexpectedError.content=Um erro inesperado ocorreu durante o processamento de (…)
alert.unexpectedError.addressee=support@company.net
mail.custom1.subject= Assunto de um e-mail personalizado pelo cliente
mail.custom1.content=Corpo do e-mail personalizado pelo cliente

Este módulo recebe via HTTP Rest POST uma solicitação para enviar um determinado e-mail. O destinatário pode ou não ser enviado junto à solicitação. No e-mail de criação de um usuário, por exemplo, o endereço do destinatário é dinâmico, dependendo do e-mail cadastrado para o usuário. Já no e-mail de alerta de erro, o destinatário é fixo, e é definido como um outro atributo no arquivo de propriedades.

5.2.3 Integração uMov.me

O módulo de integração com a plataforma uMov.me foi o escolhido por ser a integração mais comum implementada para os produtos. Todos eles dão suporte a esta plataforma, e sua utilização é incompleta sem ela, pois é através dela que são criadas as tarefas a serem realizadas em campo.

A estrutura do módulo de integração uMov.me é a mesma de todos os módulos de integração, como foi visto na Figura 17.

6 EXEMPLOS DE FLUXO

Nesta sessão serão detalhados dois exemplos de fluxos completos de execução do G4 WORK. No primeiro caso será exemplificada a criação de um usuário do tipo Executor, onde as informações do mesmo são enviadas às plataformas de mobilidade. O segundo caso mostra a recuperação de informações vindas dos dispositivos móveis através da plataforma uMov.me, até a entrada das mesmas no sistema.

6.1 Criação de Usuário Executor

A criação de um usuário do tipo Executor exemplifica diversos fluxos já mostrados. Ela inicia na validação dos dados de um formulário (Figura 18), passa pelo armazenamento de informações no banco de dados, e pelo envio de informações para uma plataforma de mobilidade (Figura 8).

Passo 1 – Validação dos dados, como visto na sessão 4.3.6. Caso haja erros na validação, o usuário é alertado e solicitado a alterar as informações que apresentam problemas. Este passo se repete enquanto houverem erros na validação dos dados inseridos pelo usuário. Caso não hajam erros mas hajam alertas, o usuário é alertado sobre todos eles e questionado se deseja prosseguir. Caso a resposta seja não, o processo encerra; caso seja sim o processo avança para o passo seguinte.

Passo 2 – Persistência dos dados no meio de armazenamento. Este passo pode ser de inclusão ou atualização de dados. Caso ocorra algum problema no salvamento dos dados, o usuário é alertado a respeito e o processo encerra. Caso contrário – os dados são persistidos com sucesso – o processamento continua.

Passo 3 – Se o Executor possuir acesso às plataformas móveis conforme informado pelo usuário efetuando o cadastro, o sistema prepara as informações para serem enviadas. Primeiramente os dados do Executor são transformados em uma entidade IntegrationDTO (Figura 14), sendo que cada informação a ser enviada é adicionada na lista de atributos em uma ordem específica. Também são adicionadas as dependências, se houverem, à lista de dependências. Esta entidade é então convertida para um texto XML, que é colocado no campo

“message” de uma nova entidade “MessageDTO”. Esta entidade é então enviada através de uma chamada HTTP Rest POST para o módulo de integração com a plataforma uMov.me. Aqui a aplicação devolve o controle para o usuário, e o restante acontece assincronamente.

Passo 4 – O controlador do módulo de integração transforma o conteúdo do campo “message” da MessageDTO recebida de volta para um IntegrationDTO, e o converte para uma entidade Agent, que é a entidade correspondente a um usuário na plataforma alvo. Esta conversão é feita recuperando os atributos que foram colocados na lista “attributes” da entidade na mesma ordem em que foram acrescentados.

Passo 5 – O Agent é então convertido para texto XML e colocado como a “message” de uma entidade MessageOut, que é inserida na fila de saída do módulo.

Passo 6 – Eventualmente, a tarefa agendada responsável por processar as mensagens da fila de saída do módulo recuperará a mensagem do Executor e enviará o seu conteúdo via HTTP Rest POST para uma URL da API da plataforma alvo.

Passo 7 – A plataforma então responde com o resultado da solicitação, e caso ela tenha sido bem-sucedida, o id da entidade criada na plataforma. Se a solicitação foi bem-sucedida, o status da mensagem é alterado para “Processada” e o módulo cria uma nova MessageDTO colocando na lista de parâmetros o resultado da solicitação e o id retornado e envia de volta para o módulo principal.Caso contrário o status é alterado para “Erro de Processamento” e em uma próxima tarefa agendada a mensagem será processada novamente.

Passo 8 – Finalmente, o módulo principal recebe a requisição, recupera os valores dos parâmetros e atualiza o campo idUmov da entidade MobileUserInfo associada ao usuário Executor criado.

O fluxo descrito encontra-se resumido nas Figuras 30, 31 e 32.

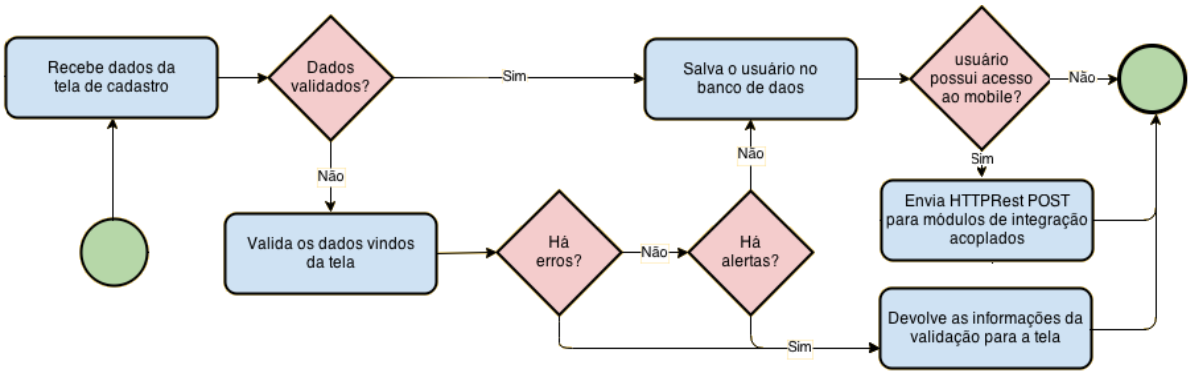

Figura 30: Envio de um Executor à plataforma uMov.me – Passos 1 a 3

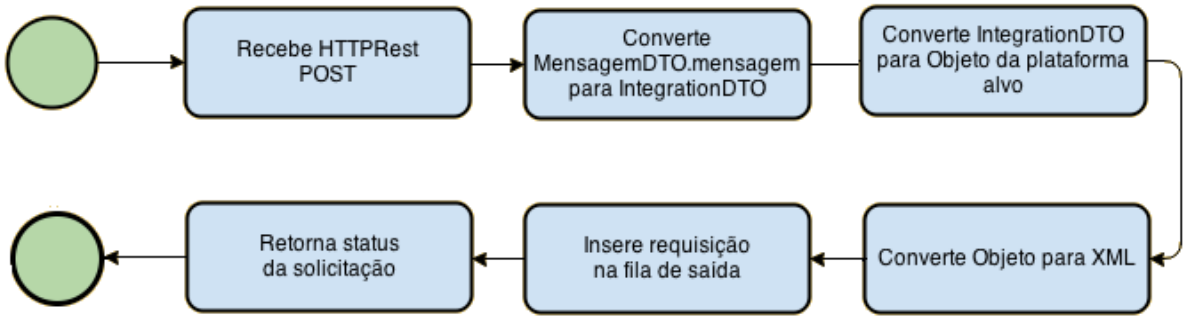

Figura 31: Envio de um Executor à plataforma uMov.me – Passos 4 e 5

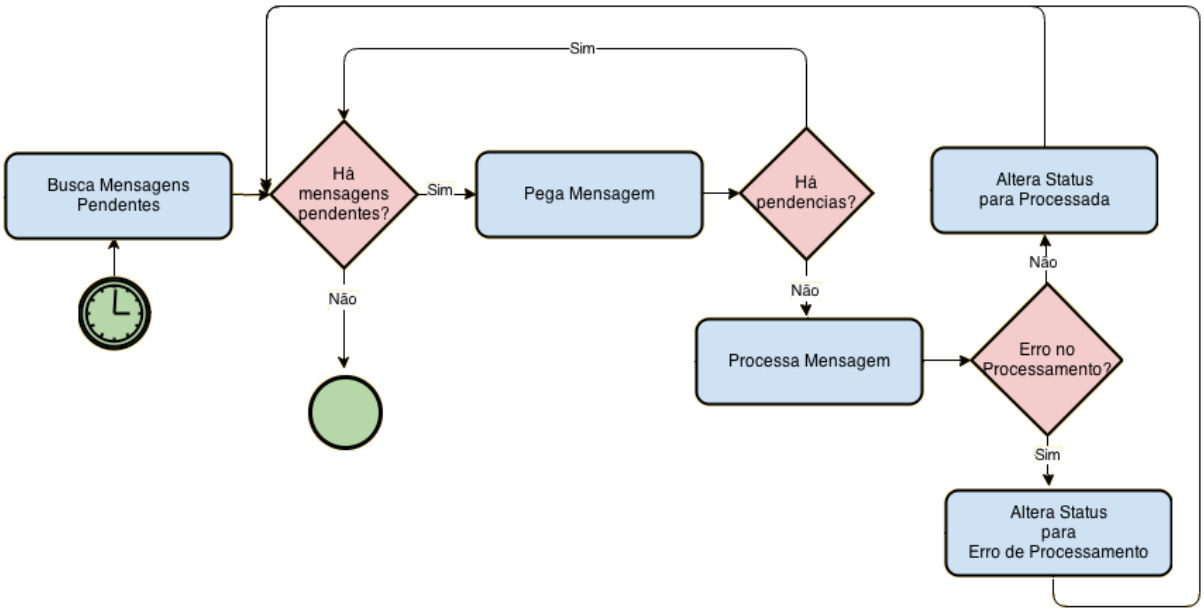

Figura 32: Envio de um Executor à plataforma uMov.me – Passos 6 e 7

6.2 Recebimento de dados do uMov.me

O recebimento dos dados vindos das plataformas de mobilidade é o fluxo mais importante dentro das soluções da X. Ele permite que os dados preenchidos pelos usuários de campo (Executores) cheguem até a sede da empresa com o menor atraso possível. Nos produtos já desenvolvidos pela X (VENDAS e SERVIÇOS), este processamento ocorre em apenas uma etapa, com o módulo de integração atualizando diretamente as entidades do módulo principal. No G4 WORK, acontece em duas etapas, sendo uma no módulo de integração com a plataforma e outra no módulo principal.

A sequência de passos executadas neste processo está descrita a seguir:

Passo 1 – O módulo de integração busca, ativamente, a partir de uma tarefa agendada, as execuções que ainda não foram importadas pelo sistema.

Passo 2 – Caso não haja nenhuma execução pendente, o processo encerra. Caso contrário, os detalhes da execução são solicitados à plataforma.

Passo 3 – Os detalhes da execução relevantes ao sistema são recebidos em formato XML e convertidos em uma entidade que modela suas características. Destas, apenas aquelas relevantes à aplicação são incluídas na lista de atributos de um IntegrationDTO.

Passo 4 – O IntegrationDTO é, então, convertido para formato XML, e inserido no campo mensagem de uma MessageDTO criada especificamente para levar os dados até o módulo principal através de uma chamada HTTP Rest POST.

Passo 5 – O módulo principal recebe o MessageDTO e insere uma mensagem "MessageIn" na sua fila de entrada com o conteúdo do MessageDTO para processamento posterior.

Passo 6 – Eventualmente, uma tarefa agendada do módulo principal recupera as mensagens pendentes na fila de entrada uma a uma e recupera o conteúdo do campo mensagem, transformando o XML novamente em IntegrationDTO.

Passo 7 – Os atributos do IntegrationDTO são então utilizados para atualizar os dados da aplicação e se tudo ocorrer corretamente, o status da mensagem é alterado para "Processada". Caso contrário o status é alterado para "Erro Processamento" e ela será processada novamente pela próxima tarefa agendada.

Os passos descritos acima são ilustrados nas Figuras 33, 34 e 35.

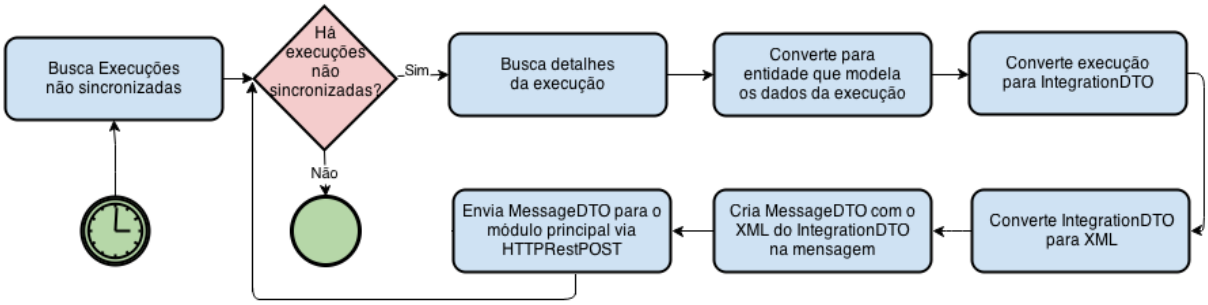

Figura 33: Recebimento de dados do uMov.me – Passos 1 a 4

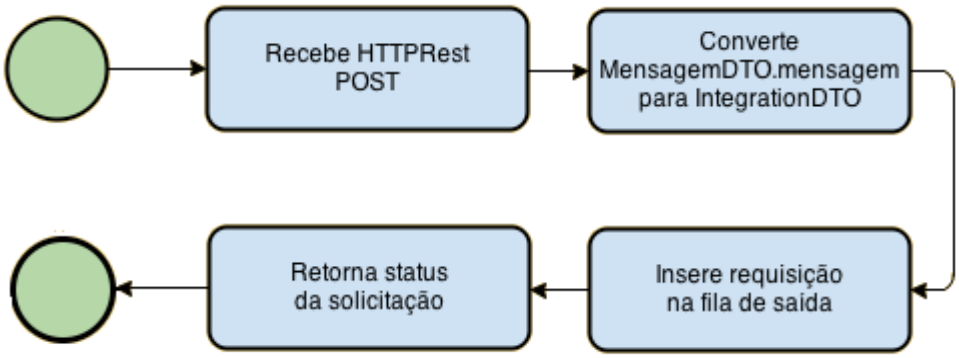

Figura 34: Recebimento de dados do uMov.me – Passo 5

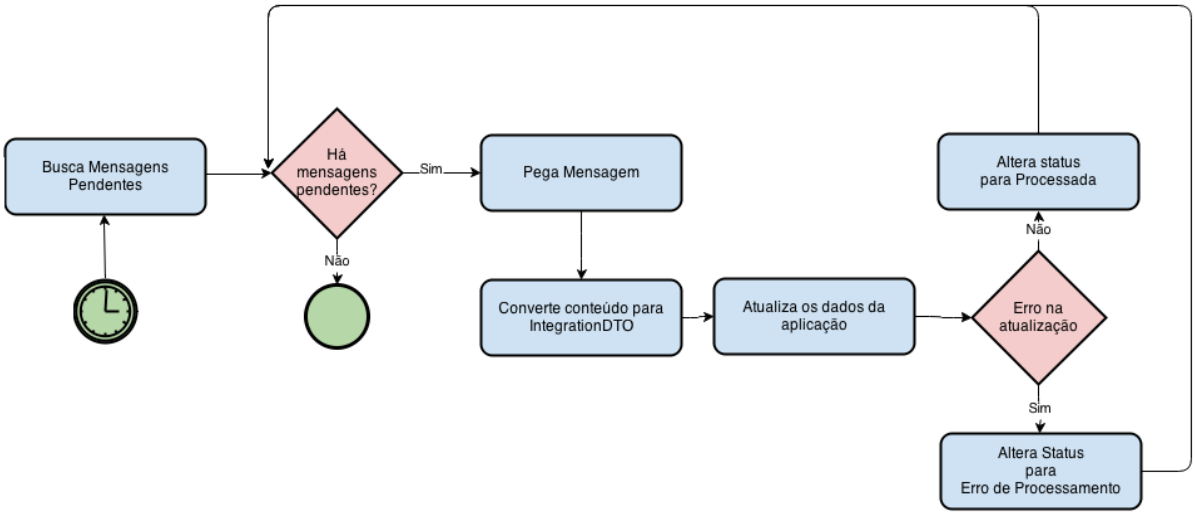

Figura 35: Recebimento de dados do uMov.me – Passos 6 a 7

7 CONCLUSÃO

Neste trabalho foram abordadas as etapas realizadas no sentido de aprimorar o processo de desenvolvimento das soluções da empresa X it solutions. Estes passos consistem na análise das soluções atuais, na proposta de uma nova topologia para as próximas soluções e na busca de possíveis melhorias a serem colocadas em prática na forma de um proto-framework. Logo após, o trabalho descreve o desenvolvimento de alguns pedaços seletos do G4 WORK, de forma a exemplificar as principais melhorias pensadas para o sistema.

Os objetivos propostos para o trabalho foram alcançados na íntegra, com a redução e códigos duplicados, a generalização de grandes processamentos e a modularização do sistema em subsistemas facilmente acopláveis. O proto-framework ainda não foi colocado em prática no desenvolvimento de soluções na empresa pois deixou muitas lacunas na parte de regras de negócios que serão preenchidas ao longo do tempo. No entanto, está claro que o G4 WORK provê uma base sólida para o desenvolvimento de novas aplicações, pois já inclui toda a parte de gerência de usuários, equipes e clientes de forma fácil e parametrizável, possibilitando o reúso e a agregação de funcionalidades de maneira isolada.

Este trabalho foi muito importante para a empresa X it solutions, pois a partir dele foi possível revelar mais do potencial que as tecnologias empregadas possuem e que não estava sendo utilizado. Também é possível estipular padrões de fluxo para as soluções novas e construir um material de consulta dinâmico e atualizado sobre o emprego e utilização do framework proposto.

Como mencionado, este trabalho dá espaço para diversos trabalhos futuros, com a implementação das regras de negócio que ficaram de fora e a extensão da flexibilização de formulários, flexibilizando a inserção de novos campos no DTO do formulário, e a criação dinâmica de código SQL para as listagens.

REFERÊNCIAS

PETERSEN, Jeremy. **Benefits of using the n-tiered approach for web applications.** , 2007. Disponível em: <http://archive.today/ua4vv#selection-1115.5-952.2>. Acesso em: 19 jun. 2014.

WEB APPLICATION framework. , 2010. Disponível em: <http://docforge.com/wiki/Web_application_framework>. Acesso em: 19 jun. 2014.

W3C. **JQuery Introduction.** Disponível em: <http://www.w3schools.com/JQuery/jquery_intro.asp>. Acesso em: 24 jun. 2014.

A. Leff; J. Rayfield "Web-application development using the Model/View/Controller design pattern". *Proc. Int. Enterprise Distrib. Object Comput. Conf.*, pp.118 -127 2001

SOLID IT (Austria) (Comp.). **DB-Engines Ranking.** Disponível em: <http://db-engines.com/en/ranking>. Acesso em: 28 jun. 2014.

GARRETT, Jesse James. **Ajax: a new approach to web applications.** , 2005. Disponível em: <https://courses.cs.washington.edu/courses/cse490h/07sp/readings/ajax_adaptive_path.pdf>. Acesso em: 19 jun. 2014.

W3TECHS (Comp.). **Usage of JavaScript libraries for websites.** Disponível em: <http://w3techs.com/technologies/overview/javascript_library/all>. Acesso em: 28 jun. 2014.