

Arnâncio Joaquim Langane

**Concepção de um Sistema de Gestão de Projectos de Construção – Estudo de caso:
Ministério da Educação na Direcção de Planificação e Cooperação de Construções e
Equipamento Escolar.
Licenciatura em Engenharia Informática**

Universidade Pedagógica
Maputo
2013

Arnâncio Joaquim Langane

**Concepção de um Sistema de Gestão de Projectos de Construção – Estudo de caso:
Ministério da Educação na Direcção de Planificação e Cooperação de Construções e
Equipamento Escolar.**

Monografia apresentada ao Departamento de
Manutenção Industrial, Escola Superior Técnica,
UP Sede, para a obtenção do grau académico de
Licenciatura em Engenharia Informática

Supervisor(a):

Msc. Claudia Jovo

Universidade Pedagógica
Maputo
2013

Índice

LISTA DE TABELAS.....	iv
LISTA DE FIGURAS.....	v
LISTA DE ABREVIATURAS.....	vi
DECLARAÇÃO.....	viii
DEDICATÓRIA.....	ix
AGRADECIMENTOS.....	x
RESUMO.....	xi
ABSTRACT.....	xii
CAPITULO I-INTRODUÇÃO.....	1
1.1. Motivação.....	2
1.2. Formulação do Problema	2
1.3. Objectivo	3
1.3.1. Objectivo Geral.....	3
1.3.2. Objectivo Específico.....	3
1.4. Importancia do Tema	3
1.5. Questões de pesquisa.....	4
1.6. Hipóteses.....	4
1.7. Estrutura do trabalho	4
1.8. Metodologia	5
CAPITULO II-REVISÃO BIBLIOGRÁFICA	7
2.1. Engenharia de Software	7
2.1.1. Objectivo da Engenharia de <i>Software</i>	7
2.2. Sistema de Informação	8
2.3. Modelagem do Sistema usando a UML	9
2.3.1. A imortancia da Modelagem.....	10
2.4. Metodologias Orientadas a Objectos.....	10
2.5. Sistema de Gestão de Base de Dados.....	11
2.6. RUP	12
2.6.1. Fases do RUP.....	13
2.6.2. Vantagens do RUP em relação a outras metodologias	15
2.7. Principais tecnologias utilizadas	15

2.7.1. Java	15
2.7.1.1. Características	16
2.7.1.2. Comparação do java com outras linguagens.....	16
2.7.2. MySQL	17
2.7.3. NetBeans IDE	18
2.7.4. Java EE	18
2.7.5. JavaServer Faces.....	18
2.7.6. PrimeFaces.....	19
2.7.7. Hibernate	19
2.7.8. GlassFish Enterprise Server	21
CAPITULO III – SISTEMA DE GESTÃO DE PROJECTOS DE CONSTRUÇÃO.....	22
3.1. Apresentação do Sistema	22
3.2. Modelagem do Sistema	23
3.2.1. Fase de Concepção / Iniciação.....	22
3.2.1.1. Requisito de Negócio.....	23
3.2.1.2. Arquitectura do Sistema.....	24
3.2.1.3. Plano de Projecto	25
3.2.2. Fase de Elaboração	25
3.2.2.1. Modelo de Casos de Uso.....	26
3.2.2.2. Diagrama de Classes	26
3.2.2.3. Modelo de Análise	27
3.2.2.4. Modelo de Implementação-Regras de Negócio	29
3.2.3. Fase de Construção	29
3.2.3.1. Diagrama de Componentes	29
3.2.3.2. Descrição do Sistema.....	36
3.2.4. Fase de Transição	41
CAPITULO IV - CONCLUSÃO	43
4.1. Futuras implementações.....	43
4.2. Consideração final.....	43
4.3. Recomendações.....	44
4.4. Referência bibliográfica	45
ANEXO	47
APÊNDICE	52

LISTA DE TABELAS

Tabela 1: Comparação do RUP e outras metodologias	15
Tabela 2: Comparação entre Java e algumas Linguagens de Programação	16
Tabela 3: Plano das Fases do projecto.....	25

LISTA DE FIGURAS

Figura 1. Função de um Sistema de Informação	8
Figura 2. Sistema de Gestão de Base Dados	12
Figura 3. Fases do RUP	13
Figura 4. Arquitectura do Sistema.....	25
Figura 5. Modelo de Casos de Uso	26
Figura 6. Diagrama de Classe.....	27
Figura 7. Diagrama de Componentes	30
Figura 8. Classe IDao	30
Figura 9. Classe DaoGeneric	31
Figura 10. Arquivo hibernate.cfg.xml	32
Figura 11. Classe Concurso (parte 1)	32
Figura 11. Classe Concurso (parte 2)	33
Figura 11. Classe Concurso (parte 3)	34
Figura 12. Classe ConcursoDao	34
Figura 13. Classe ConcursoBean (parte 1)	35
Figura 13. Classe ConcursoBean (parte 2)	35
Figura 13. Classe ConcursoBean (parte 3)	36
Figura 14. Tela Inicial do Sistema.....	37
Figura 15. Tela principal.....	37
Figura 16. Formulário de Concurso.....	38
Figura 17. Formulário de Inserção	38
Figura 18. Formulário de Actualização	39
Figura 19. Exportação por Pdf.....	40
Figura 20. Formulário de Manuais de Gestão de Empreitada	40
Figura 21. Consulta de dados	41

LISTA DE ABREVIATURAS

API - Application Programming Interface
CEE - Construções e Equipamento Escolar
CRUD - Create, Read, Update, Delete
DIPLAC - Direcção de Planificação e Cooperação
FTP - File Transfer Protocol
GUI - Graphical User Interface
HTML - HyperText Markup Language
IEEE - Institute of Electrical and Electronic Engineers
IDE - Integrated Development Environment
JDBC - Java Data Base Connectivity
JDO - Java Data Objects
JOX - Java Objects in XML
JAR - Java Archive
JDK - Java Development Kit
JSP - Java Server Pages
JPA - Java Persistence API
JRXML - Jasper Reports eXtensible Markup Language
JSF - JavaServer Faces
SE - Standard Edition
RIA - Rich Internet Application
MIS - Management Information System
MINED - Ministério da Educação
TIC - Tecnologia de Informação e Comunicação
TCP/IP - Transmission Control Protocol/Internet Protocol
OO - Object Oriented
ODBC - Open Data Base Connectivity
PDF - Portable Document Format
RUP - Rational Unified Process
RF- Requisitos Funcionais
RNF- Requisitos Não Funcionais
SGBD - Sistema de Gestão de Base de Dados

SIG - Sistema de Informações Gerenciais

SI - Sistema de Informação

EE - Enterprise Edition

SQL - Structured Query Language

UML - Unified Modeling Language

XML - eXtensible Markup Language

DECLARAÇÃO

Declaro que esta Monografia é resultado da minha investigação pessoal e das orientações do meu supervisor, o seu conteúdo é original e todas as fontes consultadas estão devidamente mencionadas no texto, nas notas e na bibliografia final.

Declaro ainda que este trabalho não foi apresentado em nenhuma outra instituição para obtenção de qualquer grau académico.

Maputo, _____ de _____ de _____

(Arnâncio Joaquim Langane)

DEDICATÓRIA

Dedico este trabalho em especial os meus pais que a todo o momento estiveram presentes nesta jornada académica que chega ao fim, por ter acompanhado todos os meus passos e apoiando em todas as horas que precisei deles. Dedico por ultimo a mim mesmo pela força, coragem e superação de vários obstáculos ao logo desta jornada académica.

AGRADECIMENTOS

Em primeiro lugar agradecer a Deus pela força e o amparo em todos os momentos da minha vida, aos meus pais, por acreditarem em mim e por todo o apoio, compreensão e tolerância em todos os meus momentos de desalento, sendo eles o motivo e a força de nunca desistir desta batalha académica.

Agradecer ao Director Nacional Adjunto Dr. Eugénio Maposse, em nome da instituição que me acolheu - DIPLAC-CEE- Direcção de Planificação e Cooperação-Construção e Equipamento Escolar, por ter aberto as portas e tornado possível a realização deste estágio profissional, dispondo das condições necessárias de trabalho ao longo do estágio.

E deixo o meu mais sincero agradecimento pela disponibilidade e paciência a Msc. Cláudia Jovo (Supervisora), na orientação deste trabalho, o Dr. Célio Sengo pelo incentivo nas aulas de Sistema de Informação e aos meus colegas da faculdade em especial o Edson António Balane muito obrigado pela troca de experiência e amizade acima de tudo.

E para findar, a minha Beberão (namorada) muito obrigado pela compreensão e amor proporcionado, e a agradecer a todos os meus amigos pelo apoio e paciência para comigo em todos os meus momentos de desespero que com a sua ajuda e o seu bom humor acabavam sempre por me contagiar, em especial o Bonifácio Muchanga, Atália Daúte Manga e Belmiro Manuel dos Santos.

RESUMO

Os sistemas informatizados permitem às empresas um maior controlo dos dados e exercem grande influência no processo da gestão das mesmas. Este trabalho consiste no desenvolvimento de um Sistema de Informação para a gestão dos dados, relativamente a Construção e Equipamento Escolar do país. O sistema objectiva automatizar os processos envolvidos no processo de Construção e Equipamento Escolar, desenvolvido em *Web*, auxiliando os funcionários no seu dia-a-dia e dar suporte à integridade da informação proporcionando informações rápidas e seguras sobre os projectos.

Palavras-chave: Desenvolvimento de Sistema, Gestão de Informação, Modelagem, Linguagem Java.

ABSTRACT

The computerized systems allow the companies to have a larger control of the business and they have great influence in the decision making process. This work is the development of an Information System for data management, for School Construction and Equipment of the country. The lens system to automate the processes involved in the Construction and Equipment School, developed in Web, assisting employees in their day-to-day support and information integrity providing fast and secure information about the projects.

Keywords: System Development, Information Management, Modeling, Java Language.

CAPITULO I-INTRODUÇÃO

O controlo eficiente de todas as actividades das empresas, sempre foi um grande problema para os seus gestores. Diante desta demanda, os *Softwares* de gestão vieram tentar suprir estas necessidades tão imediatas das empresas, como por exemplo, geração automática de relatórios e estatísticas sobre as actividades de um determinado sector da empresa. Assim, surgiram os Sistemas ou Sistema de Gestão de Informação ou ainda Sistema de Informações Gerenciais (SIG; do inglês, *management information system—MIS*) que é um Sistema de Informação, tipicamente baseado em computadores, utilizado no seio de uma empresa.

Um Sistema de Informação é composto por todos os componentes que recolhem, manipulam e disseminam dados ou informação, incluem-se tipicamente *hardware*, *software*, pessoas, sistemas de comunicação como linhas telefónicas, e os dados propriamente ditos. As actividades envolvidas incluem a introdução de dados, processamento dos dados em informação, armazenamento de ambos, e a produção de resultados, como relatórios de gestão. A DIPLAC, sendo um órgão central do Ministério da Educação responsável pela planificação, formulação de projectos e propostas de “políticas e estratégias do desenvolvimento da educação e cultura a curto, médio e longo prazo, necessita sob o ponto de vista de sua progressão, de um sistema capaz de controlar uma boa parte das actividades desenvolvidas na Direcção, de entre as quais *Planificar e controlar o desenvolvimento equilibrado da rede escolar e infra-estrutural da cultura em todo o território nacional, em conformidade com os planos de desenvolvimento económico e social do país*.

O Sistema apresentado neste trabalho é uma solução voltada para a eficientização dos processos de trabalho, virados a construção, equipamento e reabilitação de escolas públicas do país, que são ministrados pela DIPLAC-CEE, Direcção de Planificação e Cooperação - Construções e Equipamento Escolar. O Sistema impactará principalmente a área dos concursos, desde a sua abertura, a apresentação das propostas dos concorrentes, a execução ate a fase final do projecto.

1.1. Motivação

A concepção deste projecto surge pela necessidade de automatizar a gestão da informação na DIPLAC-CEE, relativamente a Construção e Equipamento Escolar, diminuir atrasos verificados para disponibilizar informação, monitorar e controlar as fases da execução dos processos, visto que na actualidade, para qualquer empresa o maior obstáculo é manter seus dados de modo que, tanto internamente e externamente estejam actualizados através de sistemas confiáveis, utilizando-se as Tecnologias da Informação e Comunicação (TIC). Para tal, as empresas procuram com sistemas de informação armazenar dados específicos e pequenas funcionalidades que devem ser automatizadas para tornar ágil o processo de tomada de decisões e disponibilizar melhores serviços aos seus clientes.

Dentro deste cenário, a DIPLAC-CEE, com o crescente número de projectos que tem levado a cabo pelo país todo, necessita de um maior monitoramento e controlo dos projectos, de modo a garantir integridade da informação, integração dos dados numa única estrutura de modo a prover resposta rápida aos pedidos da informação, flexibilidade na sua consulta, alteração, e também pela comodidade na interacção entre o usuário e o sistema.

1.2. Formulação do Problema

A Direcção de Planificação e Cooperação, adiante designada "DIPLAC" é o órgão central do Ministério da Educação responsável pela planificação, formulação de projectos e propostas de “políticas e estratégias de desenvolvimento da educação e cultura a curto, médio e longo prazo”. E para responder a gestão das actividades de construção e reabilitação de Infra-estruturas de Educação bem como desenvolver acções de coordenação e integração das actividades relativas aos projectos de construção junto das várias instituições do MINED (Ministério da Educação) assim como dos financiadores surge a DIPLAC-CEE (Construções e Equipamento Escolar), sector subordinado a DIPLAC (Direcção de Planificação e Cooperação) do MINED vocacionado a construção, reabilitação de infra-estruturas escolares, bem como o seu apetrechamento.

Actualmente a DIPLAC-CEE realiza as suas actividades de gestão, usando os métodos arcaicos (tradicionais), que ao longo do tempo vão provocando situações de extravio de processos, perda de dados ou informação, demora na execução das mesmas actividades, bem

como a falta de actualização destes, no caso em que é preciso, de modo a evitar a redundância de informação.

Constatou-se ainda, que o acompanhamento da execução das obras de construção, reabilitação e ou ampliação das escolas, não é fiável, uma vez não trazendo um ponto de situação no que diz respeito ao nível ou estágio da execução das respectivas obras. Daí a necessidade de conceção de um sistema de gestão de informação.

1.3. Objectivo

1.3.1. Objectivo Geral

- Conceber um Sistema de Gestão de Projectos de Construção na Direcção de Planificação e Cooperação – Construções e Equipamento Escolar, do Ministério da Educação.

1.3.2. Objectivo Específico

- Identificar com exactidão, as dificuldades existentes na organização através do levantamento de requisitos que permitirão a modelação do sistema e as tecnologias a empregar.
- Automatizar o processo de manipulação dos dados, respectivamente: inserção, busca e actualização.
- Propor uma solução de Gestão de Projectos de Construção na DIPLAC-CEE automatizado, que permite o controlo dos dados necessários a organização.

1.4. Importancia do Tema

O seguinte tema tem como importância a Gestão automatizada da informação relativa aos processos de trabalho virados a Construção e Equipamento das escolas públicas do país, que são ministrados pela DIPLAC-CEE, de modo a manter a integridade dos dados e sanar a redundância, visto que há extravio de processos, perda de dados ou informação, demora na sua disposição devido ao modo como esta organizada e com isso tem havido prejuízos constantes. Tendo em vista a sanar esses constantes problemas com recurso às Tecnologias de Informação e Comunicação, faz-se a análise dos problemas de forma estruturada para obter melhores resultados e melhor desempenho.

1.5. Questões de pesquisa

- Até que ponto o sistema poderá colaborar para a tomada de decisão e evitar ou minimizar os problemas potenciais relacionados com a gestão de informações?
- A utilização de um Sistema de Informação será capaz de gerir os dados no departamento de construções, para a superação do problema?
- O sistema poderá facilitar as actividades dos funcionários no armazenamento de dados, na prescrição, inserção, actualização e extracção de relatórios de modo a minimizar o tempo e aumento da produtividade?

1.6. Hipóteses

- Melhorar a gestão de dados de modo a dar suporte a integridade da informação em paralelo com o crescimento da instituição.
- Minimizar o desconforto na manipulação da informação, relativamente ao armazenamento e consulta dos dados úteis a organização.
- Garantir aos funcionários de maneira clara e comoda mais flexibilidade na altura de consulta dos dados.

1.7. Estrutura do trabalho

Este trabalho está organizado em 4 capítulos, dos quais este é o primeiro. Neste capítulo é apresentado resumidamente uma explanação geral sobre o sistema, bem como a motivação, formulação do problema, os objectivos, as hipóteses e a natureza na qual este trabalho está focado.

O Capítulo 2 apresenta a revisão bibliográfica, que fundamenta as tecnologias e os conceitos utilizados no desenvolvimento do trabalho, bem como a metodologia empregada no desenvolvimento do sistema.

No Capítulo 3 está a modelagem e a implementação obtidas como resultado da realização deste trabalho. É demonstrado também algumas funcionalidades básicas do sistema em questão, expondo de forma simples algumas de suas telas.

E por fim no capítulo 4 aborda-se as considerações finais em relação ao sistema e ao trabalho, a conclusão, recomendações e a referência bibliográfica utilizada.

1.8. Metodologia

Metodologia é a descrição precisa dos métodos, materiais, técnicas e equipamentos utilizados. Para o desenvolvimento do trabalho foi feito um estudo do actual sistema de Gestão de Informação, consulta bibliográfica e diversos materiais que abordam sobre o tema, entrevista com o pessoal do Procurement e da Direcção da Organização. Tratando-se de um Sistema Gestão de Informação, o presente trabalho comporta duas principais metodologia: a metodologia do trabalho e a metodologia da modelação do sistema.

Metodologia de trabalho

Será usada metodologia aplicada, uma vez que será implementada futuramente nas instalações da DIPLAC-CEE. A pesquisa aplicada é aquela em que o pesquisador é movido pela necessidade de conhecer para aplicação imediata de resultados. Contribui para fins práticos, visando à solução mais ou menos imediata de problemas encontrados na realidade, abordando conceitos que são cruciais para entender a necessidade da solução a implementar.

Metodologia de Modelação do Sistema

A metodologia de modelação surge na necessidade de adoptar um padrão para o seu desenvolvimento que é importante para organização, compreensão, portabilidade e qualidade do código fonte gerado de modo adequar-se ao patrão mínimo de qualidade definida na Engenharia de *Software*. E para a descrição detalhada da solução em causa, será posta em prática usando a Metodologia de Desenvolvimento de *Software* - Processo Unificado Racional - RUP (*Rational Unified Process*), de modo a modelar cada etapa do sistema em causa.

O RUP usa a abordagem da Orientação a Objectos em sua concepção e é projectado e documentado utilizando a notação UML (*Unified Modeling Language*) para ilustrar os processos em acção, aplicável a grandes equipas de desenvolvimento e a grandes projectos. O seu principal objectivo é atender as necessidades do usuário garantindo uma produção de *software* de alta qualidade.

Instrumento de colecta de dados

No intuito de colher dados fiáveis para sustentar a realização do presente trabalho de pesquisa, usou-se a observação participante, onde o pesquisador esteve inserido no campo de

estudo observando e fazendo a análise sobre os constrangimentos provocados pelo sistema actual na empresa com vista a solucionar os problemas acima mencionados.

CAPITULO II-REVISÃO BIBLIOGRÁFICA

Este capítulo apresentará os conceitos que serão usados durante o desenvolvimento deste trabalho, além dos conceitos de modelagem de sistemas utilizando a linguagem apropriada.

2.1. Engenharia de Software

O *software* de computadores é hoje uma tecnologia importante no âmbito mundial, tendo um crescimento rápido desde a década de 1950, com esse ríspido crescimento começou a ocorrer problemas relacionados a correções, adaptações, aperfeiçoamento é ainda o processo de manutenção que consome mais recurso e mais pessoas que na criação de novos *software*. Com todos esses problemas novos conceitos começaram serem criados como o da Engenharia de *Software*.

Segundo Fritz Bauer citado por Pressman (2006, pag.17), a Engenharia de *software* "é a criação e a utilização de sólidos princípios de engenharia a fim de obter *softwares* econômicos que sejam confiáveis e que trabalham eficientemente em máquinas reais". Já IEEE - Instituto de Engenheiros Eletricistas e Eletrônicos, desenvolveu uma definição mais abrangente que é " aplicação de uma abordagem sistemática, disciplinada e quantificável, para desenvolvimento, operação e manutenção do *software*, isto é a aplicação da engenharia ao *software*. Os estudos de abordagens como as de".

Todas essas definições estão inseridas no livro de *Pressman* que ressalva que a engenharia de *software* é uma tecnologia em camadas (Ferramentas, Métodos, Processo, Foco na qualidade) e a organização devem se apoiar num compromisso de qualidade.

2.1.1. Objectivo da Engenharia de *Software*

Podem ser destacados os seguintes objectivos para a Engenharia de *Software*:

- Aplicação de teoria, modelos, formalismos, técnicas e ferramentas da ciência da computação e áreas afins para o desenvolvimento sistemático de *software*;
- Aplicação de métodos, técnicas e ferramentas para a gestão do processo de desenvolvimento;
- Produção de documentação formal/semi-formal destinada à comunicação entre os

membros da equipe de desenvolvimento, bem como aos clientes e usuários.

2.2. Sistema de Informação

Um Sistema é um conjunto de elementos interconectados ou interdependentes que interagem entre si com objectivos comuns e formando um todo organizado.

Informação é um dado com significado, normalmente processado de forma a ser útil. Uma informação deve permitir responder perguntas como "quando", "quanto", "quem", "qual" e "onde" sobre alguma coisa.

O Sistema de Informação (SI) é definido como uma união de objectos inter-relacionados trabalhando acoplados para colher, restaurar, processar, guardar e distribuir informação com o propósito de simplificar o controlo, o planeamento, a organização, a análise e o processo de decisão em empresas e organizações (LAUDON, 1999).

Os SI têm função essencial na gestão de uma organização, porém apenas os SI não são satisfatórios para determinar os caminhos que uma organização deve traçar. Há factores (econômicos, sociais, políticos, etc.) ainda mais complexos que resultam em uma cadeia de variáveis em que os sistemas de informações apresentam-se como uma ferramenta a mais no procedimento de tomada de decisão.

Os sistemas de informação também requerem um feedback, que é a entrada que volta a determinados membros da organização para ajudá-los a avaliar ou corrigir o estágio de entrada.

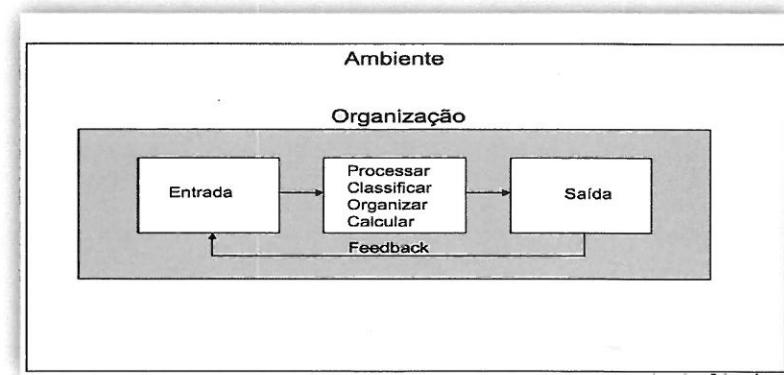


Figura 1: Função de um sistema de Informação

Fonte: <http://www.win2pdf.com>.

2.3. Modelagem de Sistemas utilizando UML

A UML (*Unified Modeling Language* - Linguagem de Modelagem Unificada) é uma linguagem de modelagem que serve para a realização de uma padronização no desenvolvimento de *software*. Sendo assim, ela prevê a elaboração e visualização de elementos existentes no desenvolvimento de *softwares* como, por exemplo, os diagramas.

Segundo Booch; Rumbaugh; Jacobson (2000), a UML é uma linguagem muito significativa, compreendendo todas as visões imprescindíveis ao desenvolvimento e implantação de sistemas de informação corporativos distribuídos a aplicações fundamentadas em Web e até mesmo sistemas complexos de tempo real. Ele acrescenta ainda, que a UML é somente uma parte de um método para desenvolvimento de *software*. Para que a realização do desenvolvimento de um projecto de *software* seja bem-sucedida, a UML estabelece alguns diagramas e padrões de desenvolvimento que tem por objectivo a especificação, a visualização, a construção e a documentação dos artefactos de um sistema de *software*.

Uma linguagem de programação fornece um vocabulário e regras para que se desenvolvam aplicações utilizando a mesma. Já uma linguagem de modelagem tem sua visão voltada para o aspecto conceptual e físico de um sistema. Portanto, uma linguagem como a UML, é uma linguagem-padrão para a preparação da composição de projectos de *software* (BOOCH, 2000).

Por meio de diagramas a UML possibilita a representação sistemas de *software* de diferentes pontos de vista. Facilitando assim a comunicação entre todas as pessoas envolvidas no desenvolvimento de um sistema como gerente, analista de sistema, coordenadores, programadores, etc. apresentando um vocabulário de entendimento facilitado. Por este motivo esta linguagem pode ser considerada uma das mais expressivas para a modelagem de sistemas orientados a objectos.

Algumas pessoas dizem que a UML é entendida como um processo de desenvolvimento de *software*. Mas na realidade ela é uma linguagem para modelagem visual, que define artefactos para a construção de *softwares* de forma a aplicá-los no processo de desenvolvimento de *software*. Ela também não pode ser considerada uma linguagem de programação visual.

A UML se destina principalmente à modelagem de sistemas de informação complexos. Actualmente esta sendo empregada nas mais diversas áreas, como por exemplo: serviço de bancos e finanças, telecomunicações, transportes, vendas, sistemas distribuídos fundamentados na Web, etc. Mas não se restringe somente ao desenvolvimento de sistemas complexos, ela pode ser utilizada no desenvolvimento de qualquer tipo de sistema.

2.3.1. A importância da Modelagem

Uma empresa de desenvolvimento de *software*, que tem por objectivo fornecer sistemas de informação com qualidade e que atendam as necessidades de seus clientes, deve realizar o desenvolvimento de um *software* de maneira previsível e em um determinado prazo. Para tanto, ela deverá utilizar de forma eficiente e eficaz os recursos disponíveis, podendo assim ser considerada uma empresa com um negócio viável.

A modelagem pode ser considerada a parte central no desenvolvimento de um bom *software*, nela se concentram todas as actividades essenciais para que isto aconteça. Através da modelagem podemos construir modelos e gráficos que explicam características ou comportamentos de um *software*. Com estes modelos pode-se visualizar e fazer o controlo da arquitectura do sistema e até mesmo realizar a gestão de riscos no desenvolvimento de um sistema de informação (BOOCH, 2000).

Segundo Booch (2000), a modelagem não se restringe somente a *software* de grande porte, os sistemas considerados de médio e pequeno porte também poderão utilizar todos os recursos disponíveis na modelagem de *software*. Isso não quer dizer que a modelagem seja menos importante em sistemas pequenos, é que quanto maior a complexidade do sistema, maior será a ênfase na modelagem do mesmo. Portanto, ainda que se considere não necessária a realização da modelagem em um sistema de *software*, à medida que ele cresce não somente no tamanho, mas também na complexidade, a equipe de desenvolvimento poderá se arrepender desta decisão.

2.4. Metodologias Orientadas a Objectos

Em Engenharia de *Software* uma *metodologia* é um conjunto estruturado de práticas (por

exemplo: Material de Treinamento, Programas de educação formais, Planilhas, e Diagramas) que pode ser repetível durante o processo de produção de *software*.

Metodologias de Engenharia de *Software* abrangem muitas disciplinas e nas principais abordagens de Metodologias de Desenvolvimento de *Software* encontramos a Metodologia Orientada a Objectos. O conceito da Orientação a Objectos (OO ou O-O, do inglês *object-oriented*) baseia-se numa nova forma de analisar o mundo. Esta abordagem reproduz a forma como o ser humano se apercebe e expressa a realidade que o rodeia. Ele classifica e subdivide o mundo em diferentes objectos, com base nas diferenças e semelhanças existentes ao nível das características e comportamento dos mesmos objectos. As técnicas orientadas por objectos identificam e definem cada Objecto de modo a reutilizá-lo, da mesma forma que o ser humano acumula conhecimento com base no previamente adquirido.

2.5. Sistema de Gestão de Base de Dados

Base de dados são coleções de informações que se relacionam de forma a criar um sentido. São de vital importância para empresas e há duas décadas se tornaram a principal peça dos sistemas de informação. A principal aplicação de Base de Dados é controlo de operações empresariais gerindo vastos conjuntos de dados de modo a facilitar a organização, manutenção e pesquisa de dados. A sua importância é visível para as organizações pois estes mantêm organizada uma série de informações relacionadas a um determinado assunto em uma determinada ordem e são operados pelos Sistemas Gestão de Base de Dados (SGBD), que se tornaram a principal peça dos Sistemas de Gestão de Informação. Um Sistema de Gestão de Base de Dados (SGBD) é usado para armazenar informação de forma que permita às pessoas examiná-la e diversas maneiras.

Um **Sistema de Gestão de Banco de Dados (SGBD)** - do inglês *Data Base Management System* (DBMS) - é o conjunto de programas de computador (*softwares*) responsáveis pela gestão de uma base de dados. Seu principal objectivo é retirar da aplicação cliente a responsabilidade de gerir o acesso, a manipulação e a organização dos dados.

O SGBD disponibiliza uma interface para que seus clientes possam incluir, alterar ou consultar dados previamente armazenados.

Os SGBD têm sete características elementares sempre observadas: Controlo de redundância,

Compartilhar Dados, Controlo de Acesso, Interfaces, Esquematização, Controlo de Integridade e *Backups*. A segunda característica (Compartilhar dados) pode ser desconsiderada principalmente em ambiente de desenvolvimento ou ainda em aplicações remotas.

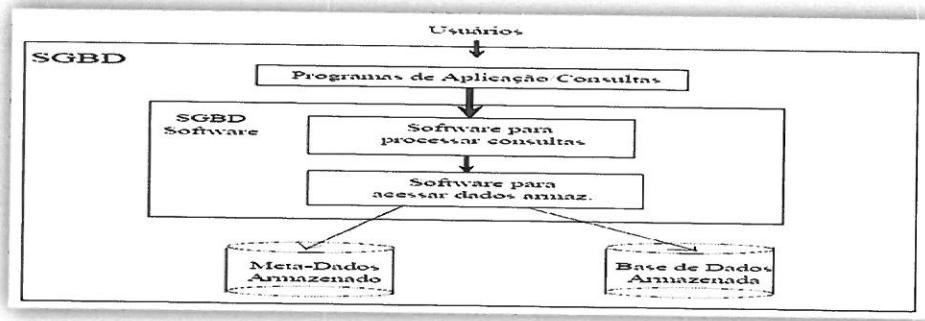


Figura 2. Sistema de Gestão de Base Dados

Fonte: <http://www.si-organizacoes.blogspot.com>

2.6. RUP

O RUP, abreviação de Rational Unified Process (ou Processo Unificado da Rational), é um processo proprietário de Engenharia de *software* criado pela *Rational Software Corporation*, adquirida pela IBM tornando-se uma marca na área de Software, fornecendo técnicas a serem seguidas pelos membros da equipe de desenvolvimento de software com o objectivo de aumentar a sua produtividade no processo de desenvolvimento.

O RUP usa a abordagem da orientação a objectos em sua concepção e é projectado e documentado utilizando a notação UML (*Unified Modeling Language*) para ilustrar os processos em acção. Dentre as suas características principais destacam-se:

- **Iterativo e incremental:** O ciclo de vida do produto é dividido em iterações, que são passagens sequenciais pelas disciplinas de engenharia de *software*. O problema total a ser resolvido é dividido em partes menores e a cada incremento uma parte acabada do *software* é entregue.
- **Guiado por casos de uso (use cases):** No caso do RUP, tal como noutras metodologias que recorrem ao UML, a inovação reside no facto dos casos de utilização, para além de especificarem os requisitos do sistema, conduzirem também o seu próprio desenho implementação e teste, ou seja, todo o processo de desenvolvimento. É o documento que conecta todas as fases e disciplinas do RUP de uma forma ou de outra.

- **Baseado na arquitectura do sistema:** É a macroestrutura que organiza os principais elementos do seu sistema, como classes, interfaces e componentes. A arquitectura evolui de acordo com as principais necessidades do sistema. Essas necessidades estão mapeadas nos casos de uso.

2.6.1. Fases do RUP

O RUP organiza o desenvolvimento de *software* em fases, onde são tratadas questões sobre planeamento, levantamento de requisitos, análise, implementação, teste e implantação do *software*. Cada fase tem um papel fundamental para que o objectivo seja cumprido, distribuído entre vários profissionais como o Analista de Sistema, Gestor de Projecto e outros.

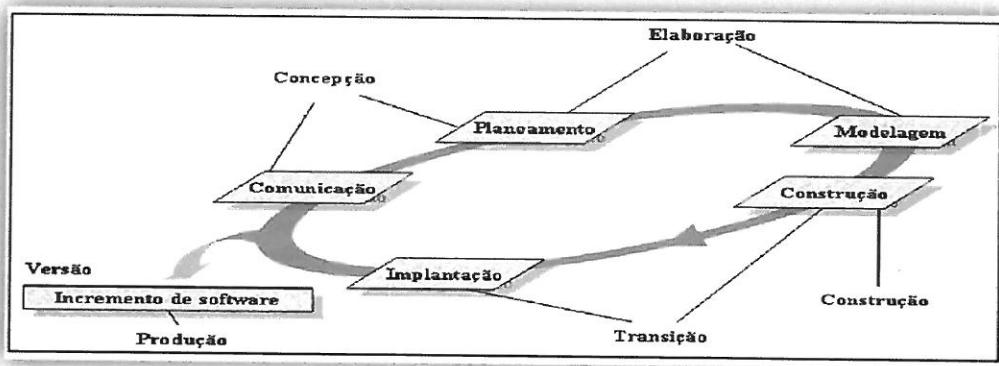


Figura 3. Fases do RUP

Fonte: PERSSMAN, Roger S. (2006,Pág 52).

Segundo (PERSSMAN, Pág.53), o RUP organiza o desenvolvimento de *software* em 5 fases, mas os importantes produtos (resultados) de trabalho são produzidos em consequência das 4 fases técnicas do RUP (PERSSMAN, Pág.54), sendo excluída a última fase, ideia esta comungada por vários outros autores que defendem que as fases do RUP são apenas 4. A fase de produção não é vista como importante para a modelagem do sistema, visto que, esta fase ocorre após disponibilizar-se o sistema, tornando-o disponível e compreendido pelo usuário final (fase de transição). Esta fase presa apenas em observar-se como o *software* é usado e os relatórios de defeito e as solicitações de modificações são submetidos e avaliados para o incremento do *software* (versão). E neste trabalho abordaremos os principais resultados apenas das 4 fases desta metodologia nomeadamente a Concepção, Elaboração, Construção e Transição, a detalhes no capítulo 3 deste trabalho, na Modelagem do Sistema.

Fase de Concepção ou Iniciação: Esta fase abrange as tarefas de comunicação com o cliente e planeamento, é feito um plano de projecto avaliando os riscos, custos e prazos, estabelecendo prioridades, levantamento dos requisitos do sistema e preliminarmente analisá-lo. Em geral, um caso de uso descreve uma sequência de acções que são realizadas por um ator à medida que o actor interage com o *software*.

Fase de Elaboração: É feita a comunicação com o cliente e actividades de modelagem do modelo genérico do processo. O objectivo desta fase é analisar de forma mais detalhada a análise do domínio do problema, refinar e expandir os casos de uso preliminares que foram desenvolvidos como parte da fase de concepção.

Fase de Construção: Desenvolve ou adquire os componentes de *software* que vão tornar cada caso de uso operacional para os usuários finais. O principal objectivo desta fase é a construção do sistema de *software*, como foco no desenvolvimento de componentes e outros recursos do sistema. É na fase de construção que maior parte de codificação ocorre.

Fase de Transição: Faz-se a entrega do *software* ao cliente para testes e relatórios de feedback sobre os defeitos e modificações necessárias. O objectivo desta fase é disponibilizar o sistema, tornando-o disponível e compreendido pelo usuário final.

Fase de Produção: Esta fase coincide com a actividade de implantação do processo genérico. Durante esta fase, o uso do *software* é monitorado, é fornecido suporte para o ambiente de operação e os relatórios de defeito e solicitações de modificações são submetidos e avaliados. (PERSSMAN, Roger S. Pág. 53).

2.6.2. Vantagens do RUP em relação a outras metodologias

Tabela 1: Comparação do RUP e outras metodologias.

Fonte: PERSSMAN, Roger S. (2006, Pág. 55).

Acção	Cascata	Modificada	Modelo V	Espiral	RUP	Ágil
Lidar com requisitos mal entendidos	NA	NA	NA	A	A	A
Lidar com arquitectura mal entendida	NA	NA		A	A	A
Produto altamente escalável	A	A	A	A	A	A
Boa gestão de riscos	NA	CR	CR	A	A	A
Ao encontro do cronograma	A	A	A	CR	A	A
Baixa sobrecarga (Sem gestão de projectos)	NA	NA	NA	CR	CR	A
Fazer correções no meio do projecto	NA	NA	CR	CR	A	A
Visibilidade no progresso dos clientes	NA	NA	CR	CR	A	A
Visibilidade no progresso para o gestor	NA	NA	CR	A	A	A
Fazer correções no meio do projecto	CR	CR	CR	A	A	A
Forte capacidade de monitoramento	A	A	A	A	A	NA
Legenda:						
A-Apropriada						
NA-Não apropriada						
CR-Com risco						

2.7. Principais tecnologias utilizadas

O sistema desenvolvido neste trabalho não deverá gerar qualquer custo para a referida empresa ou terceiros, sendo assim, consistirá em um sistema *freeware* desenvolvido inteiramente com a utilização de Tecnologias Livres relacionadas a seguir:

2.7.1. Java

Java é uma linguagem de programação orientada a objecto, desenvolvida na década de 90, pela Sun Microsystems. É considerada uma linguagem robusta que roda em vários tipos de

plataforma, não limitando o programador em somente algumas plataformas. A linguagem é composta por símbolos e palavras reservadas, usadas para escrever expressões, instruções métodos, classes, etc. Existem ainda vários tipos de IDEs (Integrated Development Environment - Ambiente de Desenvolvimento Integrado) que utilizam a linguagem Java como padrão (SANTOS, 2004).

2.7.1.1. Características

A linguagem Java foi projectada tendo em vista os seguintes objectivos:

- Orientação a Objectos;
- Portabilidade - Independência de plataforma;
- Recursos de Rede - Possui extensa biblioteca de rotinas que facilitam a cooperação com os protocolos TCP/IP, como HTTP e FTP
- Segura - Pode executar programas via rede com restrições de execução, etc.

2.7.1.2. Comparação do java com outras linguagens

Tabela 2: Comparação entre Java e algumas Linguagens de Programação

Fonte:http://pt.wikipedia.org/wiki/Anexo:Compara%C3%A7%C3%A3o_entre_linguagens_de_programa%C3%A7%C3%A3o

Linguagem	Modelo de execução	Influências	Paradigma principal	Modelo de tipo de dados	Introdução
C	Compilação	Algol, BCPL	Estruturada, Procedimental, Orientada por fluxo	Estático, fraco	Início de 1970
C++	Compilação	C, Simula, Algol 68	Principalmente orientada a objectos, múltiplos paradigmas	Estático, fraco	1979
Objective-C	Compilação	C, Smalltalk	Principalmente orientada a objectos, Reflectiva, Passagem de mensagens	Dinâmico e estático, fraco	1986

Python	Interpretação	ABC, Perl	Orientada a objectos	Dinâmico, forte	1990
Ruby	Interpretação	Smalltalk, Perl	Orientada a objectos	Dinâmico, forte	1995
Mathematica	Interpretação	LISP	Múltiplos paradigmas	Dinâmico, forte	1986
C#	Interpretação e Compilação	Java, C++	Orientada a objectos, múltiplos paradigmas	Estático, forte	2002
Java	Interpretação e Compilação	C++	Orientada a objectos	Estático, forte	1996
Perl	Interpretação	C, Shell, awk, sed, Lisp	Funcional, Orientada a objectos e Procedural	Dinâmico	1987
Boo	Interpretação	Python	Orientada a objectos	Estático	2003
PHP	Interpretação	C e Perl	Orientada a objectos	Dinâmico	1995

2.7.2. MySQL

O MySQL é um Sistema de Gestão de Base de Dado relacional (SGBD). O sistema funciona como um servidor robusto de base de dados SQL (*Structured Query Language*) multitarefa e multiusuário (MANUAL, 2013).

O MySQL é frequentemente utilizado em projectos de *software* livre que requerem um sistema de gestão de dados completo. Dentre principais características destacam-se:

- Portabilidade (suporta praticamente qualquer plataforma actual);
- Compatibilidade (existem drivers ODBC, JDBC e .NET e módulos de interface para diversas linguagens de programação, como Delphi, Java, C/C++, Python, Perl, PHP e Ruby);
- Excelente desempenho e estabilidade;
- Pouco exigente quanto a recursos de *hardware*;
- Facilidade de uso;
- É um *Software* Livre;

Uma grande vantagem é a de ter código aberto e funcionar em um grande número de sistemas operacionais: Windows, Linux, FreeBSD, BSDI, Solaris, Mac OS X, SunOS, SGI, etc.

2.7.3. NetBeans IDE

O NetBeans IDE é um ambiente de desenvolvimento integrado (IDE) gratuito e de código aberto para desenvolvedores de *software* nas linguagens Java, C, C++, PHP, Groovy, Ruby, entre outras. Foi iniciado em 1996, na Universidade de Charles, em Praga, quando a linguagem de programação Java ainda não era tão popular como actualmente.

Toda a estrutura principal do sistema neste trabalho, deverá ser desenvolvido com o uso desta plataforma visto que é um IDE executado em muitas plataformas, como Windows, Linux, Solaris e MacOS e oferece aos desenvolvedores ferramentas necessárias para criar aplicativos profissionais virados a Web.

2.7.4. Java EE

Java Enterprise Edition (Java EE) é uma plataforma padrão de desenvolvimento de aplicações para empresas, codificada na linguagem de programação Java. Com base no sólido fundamento da Plataforma Java, Standard Edition (Java SE), Java EE acrescenta bibliotecas e serviços do sistema que oferecem suporte à escalabilidade, acessibilidade, segurança, integridade e outros requisitos de aplicações de classe empresarial. (ORT, 2013).

Desde seu lançamento em 1999, Java EE amadureceu de modo a tornar-se uma plataforma funcionalmente rica e de alto desempenho. Os recentes lançamentos da plataforma sublinharam igualmente a simplicidade e facilidade de uso. Na verdade, com a plataforma Java EE, o desenvolvimento de aplicação corporativa Java nunca foi mais fácil ou mais rápido. A última versão, Java EE 6, acrescenta um significativo conjunto de novas tecnologias e amplia as melhorias de usabilidade feitas em Java EE de versões anteriores (ORT, 2013).

2.7.5. JavaServer Faces

JavaServer Faces (JSF) é um *framework* baseado em Java Web. Como estrutura de aplicação, destina-se a simplificar a integração de desenvolvimento de interfaces de usuário e das

operações que os usuários realizam no desenvolvimento com Java EE (JAVASERVER, 2013). A implementação de referência JSF é livre para usar e redistribuir.

A tecnologia *JavaServerFaces* simplifica a construção de interfaces com o usuário para aplicações *JavaServer*, isso porque os desenvolvedores podem construir aplicações *web* por meio da composição de componentes de interface com o usuário reusáveis, conectar esses componentes com uma base de dados e vincular os eventos gerados no cliente com os manipuladores de eventos que estão no servidor.

2.7.6. PrimeFaces

PrimeFaces é uma suíte de componentes de código fonte aberto para Java Server Faces 2.0 com um conjunto de mais de cem componentes JSF para o desenvolvimento de interfaces ricas (PRIMEFACES, 2013). Dentre as características de PrimeFaces, destacam-se (PRIMEFACES, 2013):

- Possui um conjunto de componentes de interface gráfica para implementação de *Rich Internet Application* (RIA), como, por exemplo, *DataTable*, *AutoComplete*, *HtmlEditor* e *Charts* (gráficos).
- Não há necessidade de configuração XML extra e não há dependências requeridas de outras tecnologias.
- A construção de Ajax é baseada no padrão JSF 2.0 Ajax APIs (*Application Programming Interface*).
- Possui um *skinning Framework* com mais de 25 temas.

2.7.7. Hibernate

O *Hibernate* é um *framework* para o mapeamento objecto-relacional escrito na linguagem Java e é também disponibilizado em *.Net* com o nome *NHibernate*. Esse *framework* facilita o mapeamento dos atributos entre uma base de dados relacional e o modelo de objectos de uma aplicação por meio do uso de arquivos XML. O *Hibernate* visa minimizar a complexidade dos programas Java baseados no modelo orientado a objecto que trabalham com base de dados no modelo relacional (HIBERNATE, 2013).

O *Hibernate* transforma os dados tabulares de uma base de dados em um grafo de objectos

definido pelo desenvolvedor. Sua principal característica é a transformação das classes Java para tabelas de dados e dos tipos de dados Java para tipos SQL (*Structured Query Language*). O *Hibernate* gera as sentenças SQL, mantendo o programa portável para qualquer base de dados padrão SQL.

Em aplicações construídas para serem executadas em servidores, a gestão das transações é realizado segundo o padrão JTA (*Java Transaction API*). Nas aplicações *desktop*, o tratamento transacional é realizado pelo *driver JDBC* (*Java Database Connectivity*).

Definições básicas:

Um *framework* (ou arcabouço), em desenvolvimento de *software*, é uma abstração que une códigos comuns entre vários projectos de *software* provendo uma funcionalidade genérica. Vulgarmente um *framework* é um conjunto de classes que colaboram para realizar uma responsabilidade para um domínio de um subsistema da aplicação.

JDBC-Java Database Connectivity é um conjunto de classes e Interfaces (API) escritas em Java que fazem o envio de instruções SQL para qualquer base de dados relacional. JDBC é uma biblioteca vinculada a API da linguagem Java que define como um cliente pode aceder uma base de dados. A API JDBC já está vinculada ao JDK do Java, porém é necessário baixar as APIs específicas para o Oracle que são conhecidas como *connectors*. O *connector* é a implementação das interfaces JDBC. Ela possui a extensão JAR e deve ser adicionada ao projecto que a utilizará.

Java Development Kit (JDK) significa **Kit de Desenvolvimento Java**, e é um conjunto de utilitários que permitem criar sistemas de *software* para a plataforma Java. É composto por compilador e bibliotecas.

Java Archive (JAR) é um arquivo compactado usado para distribuir um conjunto de classes Java, um aplicativo java, ou outros itens como imagens, XMLs, entre outros. É usado para armazenar classes compiladas e metadados associados que podem constituir um programa.

API é o acrônimo de *Application Programming Interface* ou, em português, Interface de Programação de Aplicativos. Esta interface é o conjunto de padrões de programação que permite a construção de aplicativos e a sua utilização de maneira não tão evidente para os usuários. Ela funciona através da comunicação entre diversos códigos, definindo assim

comportamentos específicos de determinados objectos em uma interface, ou seja, a API irá interligar diversas funções em um site (por exemplo, busca de imagens, notícias, artigos, etc.) de modo a possibilitar que possam ser utilizadas em outras aplicações.

2.7.8. GlassFish Enterprise Server

Segundo Ort (2006), o projecto GlassFish é um ambiente estruturado para o desenvolvimento de um servidor de aplicativos de código aberto. A criação do projecto GlassFish é parte de um esforço maior da empresa Sun Microsystems para a liberação de grandes porções do código fonte do servidor, implicando em código aberto para uso e aproveitamento da comunidade. O projecto GlassFish é a concepção de uma variedade de projectos com base na sua comunidade de desenvolvedores e está hospedado no java.net.

O Sun GlassFish Enterprise Server é um servidor de aplicativos de código aberto, que a comunidade construiu dentro do projecto GlassFish. Este servidor de aplicativos de código aberto é baseado no código-fonte do Sun Java System Application Server Platform Edition 9, doado pela Sun Microsystems, e no código-fonte para TopLink, um framework de mapeamento objecto-relacional, doado pela Oracle. A comunidade GlassFish, periodicamente faz compilações do servidor open source de aplicativos disponíveis para download. O servidor GlassFish coloca uma fonte livre, aberta, comercial, ao nível de implementação do Java EE, nas mãos da comunidade (ORT, 2006).

CAPITULO III – SISTEMA DE GESTÃO DE PROJECTOS DE CONSTRUÇÃO

3.1. Apresentação do Sistema

Para melhor concretizar os objectivos da pesquisa realizou-se um estudo de caso. Sendo assim o nosso campo de estudo foi a DIPLAC-CEE, que é um órgão central do Ministério da Educação localizada na rua Joe Slovo, antiga rua Joaquim Lapa nº 22, na cidade de Maputo.

A DIPLAC-CEE é constituída por 4 repartições que nomeadamente: Construções, Procurement, Contabilidade e Administração Interna. Mas para o presente trabalho interessam-nos a repartição das Construções, que é responsável pelo processo das Construção e Equipamento Escolar. A instituição é composta por 34 trabalhadores dentre os quais 10 elementos fazem parte da repartição das Construções (população em estudo). Sendo assim a nossa amostra é de 50% da população em estudo, uma vez que foram por nós entrevistados 5 elementos. Desta feita considera-se que a amostra é significativa pois é maior que 30%. No acto da análise dos dados recolhidos na entrevista recorremos que é um Sistema de Gestão de Projectos de Construção que abaixo fazemos a sua descrição de modo a sanar o problema retro mencionado:

O acesso ao Sistema será feito por meio de login, como ilustra a figura 14. O cadastro dos usuários com as respectivas permissões será realizado pelo administrador do Sistema. Após a autenticação, se o usuário e a senha forem correcto o sistema o direcionará para a página principal do sistema, como ilustra a figura 15.

Fase da abertura de concurso: O Coordenador e ou Administrador registam o Concurso junto com os respectivos Lotes no Sistema. Após isso, o Concorrente submete a sua proposta na Secretaria, onde é feito registo de entrada. Segue-se então, a abertura do Concurso onde são abertas as Propostas submetidas pelos Concorrentes e feito o registo das mesmas que vêm seladas e abertas na presença de todos os concorrentes. Vista a abertura e feito o registo das propostas segue-se a avaliação e o registo das mesmas.

O Concorrente faz a solicitação da avaliação das propostas e o Coordenador emite a solicitação da mesma. Nesta impressão faz-se menção as posições ocupadas pelos Concorrentes de acordo com as percentagem obtidas na avaliação. O Sistema permitirá também o registo do ponto de situação de um Concurso, o que facilitará a fazer o controlo. Após o fecho da primeira fase referente a abertura de Concurso, segue-se a execução do

projecto e o Coordenador fará o controlo do processo de Inserção, Actualização, Exclusão, Consulta e Impressão da informação que será requerida dos processos.

3.2. Modelagem do Sistema

Para a concepção deste sistema, foi escolhido o RUP para especificar os resultados obtidos em cada fase de desenvolvimento e esta escolha deve-se as vantagens do RUP em relação a outras metodologias mencionados na página 15 e pelo facto do autor ter-se familiarizado com esta metodologia ao longo do curso na cadeira de Sistema de Informação.

Para a materialização do sistema usaram-se ferramentas descritas abaixo. E referir que a escolha destas ferramentas deve-se ao facto de serem do domínio do autor e por alguns pontos mencionados no capítulo 2 referente a Revisão bibliográfica.

- *NetBeans IDE 7.1.2 (Integrated Development Environment)*: plataforma para o desenvolvimento do sistema.
- Linguagem Java: para a construção do código.
- JavaServer Faces 2.0: para controlo entre as classes Java e os JSPs.
- PrimeFaces: para desenvolvimento da interface.
- Hibernate: *framework* para o mapeamento objecto relacional.
- Glassfish: como contêiner/servidor *web*.
- Mysql: para o sistema de Gestão de Base de Dados

3.2.1. Fase de Concepção / Iniciação

Esta fase do RUP abrange as tarefas de comunicação com o cliente e planeamento. É feito um plano de projecto, estabelecendo as prioridades, levantamento dos requisitos do sistema e preliminarmente analisá-lo. Os principais resultados a obter no final desta fase são: Requisitos do Negócio, Arquitectura do Sistema e o Plano do Projecto que vai ser seguido.

3.2.1.1. Requisito de Negócio

Requisito de Negócio é uma actividade que deve ser executada em conjunto (envolvidos na construção do sistema), de modo que seja produzido um artefacto “visão de negócio”. Para

tal, faz-se uma descrição de como o sistema deve se comportar, tendo em conta o objectivo do negócio.

No caso concreto, a DIPLAC-CEE ainda se depara com o problema de Gestão de Informação, uma vez que esta é feita utilizando-se os métodos arcaicos (manualmente), não sendo de carácter confiável porque os dados são guardados em forma de ficheiros organizados, sendo desta forma susceptíveis a sua manipulação, redundância de informação, perda ou extravio dos mesmos devido a forma dos arquivos.

Neste cenário, a DIPLAC-CEE entende que as Tecnologias de Informação e Comunicação desempenham um papel relevante. Sendo a informação, um recurso importante para a organização, ela necessita de uma gestão. Esta gestão, por sua vez, necessita de um Sistema de Gestão de Informação (Sistema automatizado) para a operacionalização da gestão organizacional, respondendo desta forma o problema retro mencionado.

3.2.1.2. Arquitectura do Sistema

A Arquitectura de um Sistema reflete as grandes decisões a respeito da implementação. Diz em quantas Camadas o Sistema é feito. Arquitectura do presente Sistema é Local-Intranet composta pela Camada de Apresentação, Negócios e de Persistência.

Camada de Apresentação é de GUI (Graphical User Interface), ou simplesmente interface, interage directamente com o usuário, é através dela que são feitas as requisições (por exemplo: consulta).

Camada de Negócios também chamada de Logica empresarial, Regras de Negócio ou Funcionalidade, é nela que ficam as funções e regras de todo o negócio. Não existe uma interface para o usuário e os seus dados são voláteis, ou seja, para que algum dado seja mantido deve ser utilizada a Camada de dados.

Camada de Persistência também conhecida como de Dados é definida como repositório das informações e as classes que a manipulam. Esta camada recebe as requisições da Camada de Negocio e seus métodos executam essas requisições em uma base de dados. Alterando a base de dados alteraria apenas as classes da camada de dados, e o restante das camadas não seriam afetados por essa alteração.

Segundo o RUP a Arquitectura do Sistema inclui as decisões significativas sobre a organização de um sistema. A seleção dos elementos estruturadores e suas interfaces, a especificação dos elementos do sistema e como eles colaboram entre si. Como ilustra a figura 4.

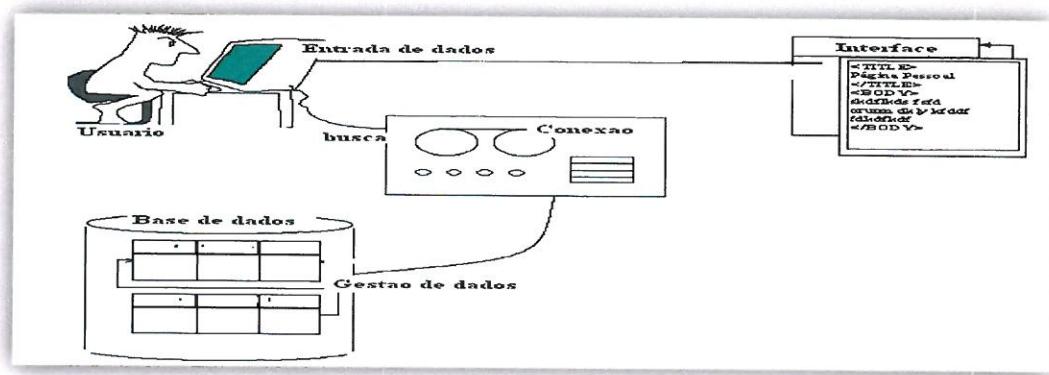


Figura 4: Arquitectura do Sistema (O autor)

3.2.1.3. Plano de Projecto

Neste ponto, apresenta-se para cada fase, uma estimativa das datas de início e fim, bem como dos principais pontos de controlo.

Tabela 3: Plano das Fases do Projecto (O autor)

Fases	Duração	Esforço
Concepção	10%	5%
Elaboração	30%	20%
Construção	50%	65%
Transição	10%	10%

3.2.2. Fase de Elaboração

Abrange a Modelagem do modelo genérico do processo. Tem por objectivo especificar as funcionalidades. Alguns dos principais resultados a obter nesta fase o Modelo de Casos de Uso, Modelo de dados (Diagrama de classe), Modelo de análise (Especificificar Requisitos) e Modelo de implementação (Regras do negócio).

3.2.2.1. Modelo de Casos de Uso

Um caso de uso é uma técnica de modelagem usada para descrever o que um sistema deve fazer. Ele é construído através de um processo interativo no qual as discussões entre o cliente

e os desenvolvedores do sistema conduzem a uma especificação do sistema da qual todos estão de acordo. E tem por objectivo decidir e descrever os requisitos funcionais do sistema que será automatizado e fornecer uma descrição clara e consistente do que o sistema deve fazer.

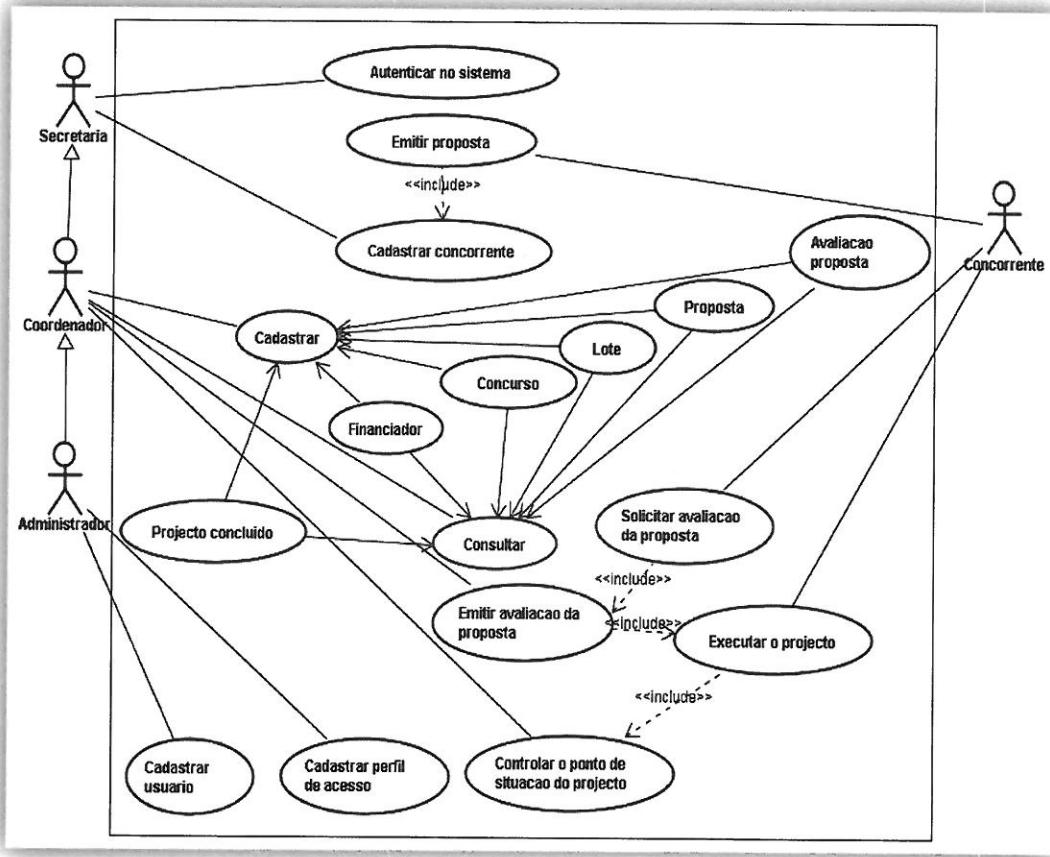


Figura 5: Modelo de Casos de Uso (O autor)

3.2.2.2. Diagrama de Classes

Diagrama de classe é uma modelagem de dado muito útil, pois define todas as classes que o sistema necessita possuir. Uma classe representa um conjunto de objectos que possuem comportamentos e características comuns. O Diagrama de Classes apresentado na fig. 6 modela a base de dados do Sistema de Gestão de Projectos de Construção na DIPLAC, exibindo todas as entidades e seus atributos que devem ser persistidos. E esta persistência foi feita com uso do framework *Hibernate*.

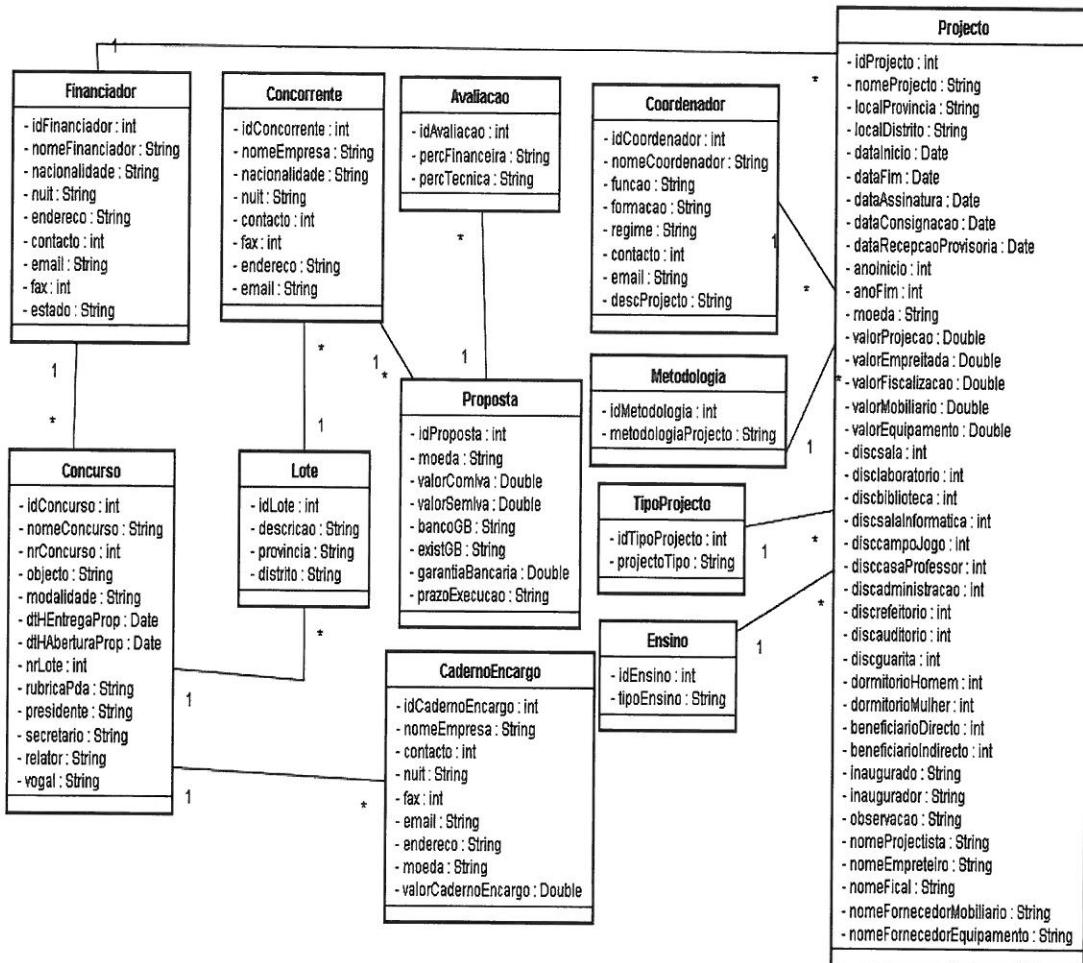


Figura 6: Diagrama de classes (O autor).

3.2.2.3. Modelo de Análise

Modelo de Análise ou Especificação de Requisitos tem como objectivo obter produtos de software de melhor qualidade que satisfaçam às reais necessidades dos clientes. Especificar um requisito implica compreender exatamente o que deve ser feito e que se espera receber como resultado. Podemos classificar requisitos em funcionais ou não funcionais:

a) Requisitos funcionais

Requisitos funcionais são declarações de funções que o sistema deve fornecer, como Sistema deve reagir a entradas específicas e como deve se comportar em determinadas situações:

[RF01] Efetuar login.

O sistema permitirá que os usuários acessem o sistema através de usuário e a senha.

[RF02] Inserir, Alterar, Consultar e Excluir dados.

O sistema permitirá que haja registo dos dados e posterior a isso os mesmos sejam consultados, alterados e excluídos.

[RF03] Gerar relatórios.

O sistema permitirá a geração de relatórios.

[RF03] Consultar documentação.

O sistema permitirá a localização dos documentos relacionados com Gestão de Empreitada e do ponto de situação dos projectos.

[RF04] Sair do sistema.

O sistema disponibilizará a opção para encerrar a sessão do usuário conectado.

b) Requisitos Não Funcionais

Requisitos não funcionais são restrições sobre os serviços ou as funções oferecidos pelo Sistema. Entre eles destacam-se restrições de tempo, restrições sobre o processo de desenvolvimento, padrões, entre outros. Os requisitos não funcionais descrevem atributos do sistema, bem como atributos do ambiente do sistema:

[RNF01] O sistema deverá ser implementado de uma forma que possa manter a compatibilidade com os principais navegadores utilizados.

[RNF02] O sistema deverá rodar em qualquer plataforma.

[RNF03] O sistema deverá ser desenvolvido na Linguagem Java.

[RNF04] Utilizar base de dados MySQL.

[RNF05] Os usuários do sistema deverão se utilizar de usuário e senha para usufruir as funcionalidades correlativas no sistema.

[RNF06] O tempo de resposta para qualquer operação realizada pelo sistema não deve exceder 20 segundos.

[RNF07] O sistema deverá ser capaz de suportar uma quantidade exorbitante acesso simultâneos.

[RNF08] O sistema precisará apresentar uma interface objectiva, amigável, consistente, intuitiva e de fácil acessibilidade, isto é, suas informações e funcionalidades e deverão estar bem visíveis e disponíveis.

[RNF09] Não deverá ser gerado qualquer custo para a implantação do sistema.

3.2.2.4. Modelo de Implementação-Regras de Negócio

O RUP define Regras de Negócio como declarações sobre políticas ou condições que devem ser cumpridas pelo sistema a fim de atender a um objectivo do negócio. Enquanto os requisitos de negócio definem o que o cliente deseja que o sistema faça. Já a regra de negócio define parâmetros para que o sistema possa garantir a qualidade e integridade dos dados:

- Login do Usuário: O sistema permitirá que os usuários acessem através do nome de usuário e a senha corretos, com determinados privilégios.
- Gestão de projectos de Construção: O sistema deve permitir a manipulação da informação no que concerne a Inserção, Actualização, Consulta e a Impressão da informação pelos parâmetros requeridos.

3.2.3. Fase de Construção

Nesta fase, começa o desenvolvimento físico do *software*, produção de códigos e testes. O principal objectivo desta fase é a construção do sistema de *software*, com foco no desenvolvimento de componentes e outros recursos do sistema, que vão tornar cada caso de uso operacional para o usuário final. São resultados desta fase o Diagrama de Componentes, Descrição do Sistema: Características e Funcionalidades do Sistema desenvolvido (implementação e testes).

3.2.3.1. Diagrama de Componentes

Componente é a parte física de um sistema (feita de bit e bytes) e substituível de um sistema, que proporciona a realização de um conjunto de interfaces. Ex: Executáveis, Bibliotecas, Tabelas, Ficheiros, Documentos. Um componente representa um empacotamento físico de elementos relacionados logicamente (normalmente classes).

Interface é um conjunto de operações usado para especificar os serviços de uma classe ou componente. Um componente pode implementar ou usar uma ou mais classes.

O Diagrama de Componentes mostra como as classes deverão se encontrar organizadas através da noção de componentes. É utilizado para:

- Modelar os dados do código fonte, do código executável do *software*.
- Destacar a função de cada módulo para facilitar a sua reutilização.

- Auxiliar no processo de engenharia reversa, por meio da organização dos módulos do sistema e seus relacionamentos.

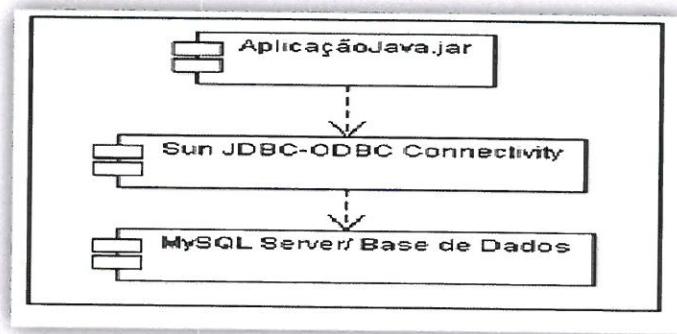


Figura 7: Diagrama de componentes

Fonte: <http://www.teccconcursos.com.br/conteudo/questoes/19725?materia=ti---engenharia-de-software&banca=cespe>.

A persistência no Sistema em causa foi feita usando-se *framework Hibernate*, descrita numa classe IDao implementada na Classe DaoGeneric, com as funcionalidades de Inserir, Actualizar, Excluir e Listar, como ilustram as figuras 8 e 9.

O código exibe a interface IDao.java, que estende a classe DaoGeneric. Ele define métodos para inserção, exclusão, atualização e merge de entidades, bem como para listar todas as entidades de uma classe específica.

```

1 package com.mz.diplac.dao;
2
3 import java.util.List;
4
5 /**
6 * Author: Langane
7 */
8 public interface IDao<T> {
9     void create(T t);
10    void delete(T t);
11    void update(T t);
12    void merge(T t);
13
14    List<T> listAll(Class<T> classe);
15    T load(Class<T> classe, Integer id);
16
17 }
18
19
20
  
```

Figura 8: Classe IDao (O autor)

```

1 package com.mz.diplac.dao.impl;
2
3 import com.mz.diplac.dao.IDao;
4 import com.mz.diplac.util.HibernateUtil;
5 import java.util.List;
6 import org.hibernate.Criteria;
7 import org.hibernate.Session;
8 import org.hibernate.Transaction;
9
10 /**
11  * Author: Lengane
12  */
13
14 public class DaoGeneric <T> implements IDao<T> {
15     private final Session session;
16     private final Transaction transaction;
17
18     public DaoGeneric() {
19         this.session=HibernateUtil.getSessionFactory().openSession();
20         this.transaction=session.beginTransaction();
21     }
22
23     @Override
24     public void create(T t) {
25         // TODO Auto-generated method stub
26         session.save(t);
27         transaction.commit();
28     }
29
30     @Override
31     public void delete(T t) {
32         // TODO Auto-generated method stub
33         session.delete(t);
34         transaction.commit();
35     }
36     @Override
37     public void update(T t) {
38         // TODO Auto-generated method stub
39         session.update(t);
40         transaction.commit();
41     }
42     @Override
43     public void merge(T t) {
44         // TODO Auto-generated method stub
45         session.merge(t);
46         transaction.commit();
47     }
48
49     @Override
50     public List<T> listAll(Class<T> classe) {
51         // TODO Auto-generated method stub
52         Criteria criteria = session.createCriteria(classe);
53         return criteria.list();
54     }
55
56     @SuppressWarnings("unchecked")
57     @Override
58     public List<T> listAll(Class<T> classe) {
59         // TODO Auto-generated method stub
60         Criteria criteria = session.createCriteria(classe);
61         return criteria.list();
62     }
63
64     @SuppressWarnings("unchecked")
65     @Override
66     public T load(Class<T> classe, Integer id) {
67         // TODO Auto-generated method stub
68         return (T) session.load(classe, id);
69     }
70
71     /**
72      * @return the session
73      */
74     public Session getSession() {
75         return session;
76     }
77
78     /**
79      * @return the transaction
80      */
81     public Transaction getTransaction() {
82         return transaction;
83     }
84
85 }
86

```

Figura 9: Classe DaoGeneric (O autor)

A conexão com a base de dados é feita usando o *Hibernate*. Estes dados geralmente são informados em um arquivo "hibernate.cfg.xml", como mostra a figura 10. Este *framework* facilita o mapeamento dos atributos de uma base de dados relacional com os objectos do sistema. Esse mapeamento é feito através de arquivos XML ou com o uso de *annotations*. Para o desenvolvimento deste sistema utilizou-se anotações.

```

1  <?xml version="1.0" encoding="UTF-8"?>
2  <!DOCTYPE hibernate-configuration PUBLIC "-//Hibernate/Hibernate Configuration DTD 3.0//EN"
3  "http://hibernate.sourceforge.net/hibernate-configuration-3.0.dtd">
4  <hibernate-configuration>
5    <session-factory>
6      <property name="hibernate.dialect">org.hibernate.dialect.MySQLDialect</property>
7      <property name="hibernate.connection.driver_class">com.mysql.jdbc.Driver</property>
8      <property name="hibernate.connection.url">jdbc:mysql://localhost:3306/bd_diplac</property>
9      <property name="hibernate.connection.username">root</property>
10     <property name="hibernate.connection.password">langane11</property>
11     <property name="hibernate.show_sql">true</property>
12     <property name="hibernate.format_sql">true</property>
13     <property name="hibernate.hbm2ddl.auto">update</property>
14     <mapping class="com.mz.diplac.model.Projeto"/>
15     <mapping class="com.mz.diplac.model.Concorrente"/>
16     <mapping class="com.mz.diplac.model.Date"/>
17     <mapping class="com.mz.diplac.model.Comissao"/>
18     <mapping class="com.mz.diplac.model.Avaliacao"/>
19     <mapping class="com.mz.diplac.model.AgenteService"/>
20     <mapping class="com.mz.diplac.model.CadernoEncarap"/>
21     <mapping class="com.mz.diplac.model.Financeiro"/>
22     <mapping class="com.mz.diplac.model.Coordenador"/>
23     <mapping class="com.mz.diplac.model.Projecto"/>
24     <mapping class="com.mz.diplac.model.PontosLancarac"/>
25     <mapping class="com.mz.diplac.model.Engene"/>
26     <mapping class="com.mz.diplac.model.Methodologia"/>
27     <mapping class="com.mz.diplac.model.TipoProjecto"/>
28     <mapping class="com.mz.diplac.model.Usuario"/>
29     <mapping class="com.mz.diplac.model.Perfil"/>
30   </session-factory>
31 </hibernate-configuration>

```

Figura 10: Arquivo hibernate.cfg.xml (O autor)

A figura 11 faz uma ilustração de uma classe pública "Concurso", entidade que faz parte do Sistema.

```

1  package com.mz.diplac.model;
2
3  import java.io.Serializable;
4  import java.util.Date;
5  import java.util.List;
6  import javax.persistence.*;
7
8  /**
9   * @author Lengana
10  */
11 @Entity
12 @Table
13 public class Concurso implements Serializable{
14   @Id
15   @GeneratedValue(strategy= GenerationType.AUTO)
16   private Integer idConcurso;
17   private String nomeConcurso;
18   private String nrConcurso;
19   private String objecto;
20   private String modalidade;
21   @Temporal(value=TemporalType.TIMESTAMP)
22   private Date dtLancConcurso;
23   @Temporal(value=TemporalType.TIMESTAMP)
24   private Date dtHEntregaProp;
25   private Integer nrLote;
26   private String rubricaFda;
27   private String moeda;
28   private Double valorEstimado;
29   private String responsavel;
30   private String presidente;
31   private String secretario;
32   private String relator;
33   private String vogal;

```

Figura 11: Classe Concurso - Parte 1 (O autor)

Pormenores:

package – Diretório (pacote) onde se encontra a classe.

import - Inclusão de outras classes na classe e bibliotecas em desenvolvimento.

@Entity - Anotação para persistência de classes em *Hibernate*.

@Id - Anotação em *Hibernate* para especificar chave primária.

@GeneratedValue - Especifica que a chave primária será gerada automaticamente.

`@TemporalType.TIMESTAMP` – Usado para mapear hora e data.

A figura 11: Classe Concurso (Parte 2), ilustra o relacionamento usando anotações com recurso a *Hibernate*.

```

33
34     @OneToMany(mappedBy="concurso", cascade=CascadeType.ALL, fetch= FetchType.LAZY)
35     public List<Lote> lotes;
36
37     @OneToMany(mappedBy="concurso", cascade=CascadeType.ALL, fetch= FetchType.LAZY)
38     public List<CadernoEncargo> cadernoEncargos;
39
40     @ManyToOne(fetch= FetchType.LAZY)
41     @JoinColumn(name="idFinanciador")
42     public Financiador financiador;
43

```

Figura 11: Classe Concurso - Parte 2 (O autor)

Pormenores:

`@OneToMany` -Anotação de mapeamento de um para muitos.

`@ManyToMany` -Anotação de mapeamento de muitos para muitos.

`@Cascade` - Anotação de mapeamento que indica com que acção em cascata o relacionamento será tratado com a entidade associada.

`@JoinColumn` -Anotação utilizada para informar qual o nome da coluna que corresponde utilizada como chave estrangeira do mapeamento.

`@fectch` – Anotação de identificação do tipo retorno dos dados.

A figura 11: Classe Concurso (Parte 3) ilustra uma parte dos métodos públicos *getters* e *setters* desta classe para que sejam acedidos fora da classe.

```

86     public Integer getIdConcurso() {
87         return idConcurso;
88     }
89
90     public void setIdConcurso(Integer idConcurso) {
91         this.idConcurso = idConcurso;
92     }
93
94     public String getModalidade() {
95         return modalidade;
96     }
97
98     public void setModalidade(String modalidade) {
99         this.modalidade = modalidade;
100    }
101
102    public String getNrConcurso() {
103        return nrConcurso;
104    }
105
106    public void setNrConcurso(String nrConcurso) {
107        this.nrConcurso = nrConcurso;
108    }
109
110    public Integer getNrLote() {
111        return nrLote;
112    }
113

```

Figura 11: Classe Concurso - Parte 3(O Autor)

DAO (Data Access Object) é um padrão de projecto utilizado em Engenharia de Software Orientados a Objectos, que se encarrega de fazer a ligação entre o *Hibernate* e as classes controladoras de cada CRUD (Create, Read, Update, Delete) da aplicação Web. No pacote com.mz.model.dao.impl, encontra-se a classe ConcursoDao como ilustra a figura 12, que herda as características da classe DaoGeneric.

```

1  package com.mz.diplac.dao.impl;
2
3
4  import com.mz.diplac.model.Concurso;
5
6  /**
7   *
8   * @author Langane
9   */
10 public class ConcursoDao extends DaoGeneric<Concurso> {
11
12 }
13

```

Figura 12: Classe ConcursoDao (O autor)

Na classe ConcursoBean temos a anotação `@ManagedBean` que serve de intermediário entre a página e o seu modelo para operacionalizar a informação da base de dados. A anotação `@SessionScoped` serve para jogar o objecto na sessão e sendo assim pode ser utilizada em diversas páginas diferentes. Como ilustra a figura 13: Classe ConcursoBean (parte 1), temos atributos a serem usados pelos métodos que serão manipulados directamente da página como ilustra a figura 13: Classe ConcursoBean (parte 2).

```

1 package com.mz.diplac.controller;
2
3 import com.mz.diplac.dao.impl.ConcursoDao;
4 import com.mz.diplac.model.Concurso;
5 import com.mz.diplac.model.Financiador;
6 import java.util.List;
7 import java.util.SortedMap;
8 import java.util.TreeMap;
9 import javax.faces.application.FacesMessage;
10 import javax.faces.bean.ManagedBean;
11 import javax.faces.bean.SessionScoped;
12 import javax.faces.context.FacesContext;
13 import javax.faces.event.ActionEvent;
14 import javax.faces.model.DataModel;
15 import javax.faces.model.ListDataModel;
16
17 /*
18 * @author Langane
19 */
20 @ManagedBean
21 @SessionScoped
22 public class ConcursoBean {
23
24     private Concurso concurso;
25     private DataModel listDM;
26     private ConcursoDao dao;
27     private List<Concurso> filteredConcursos;

```

Figura 13: Classe ConcursoBean - parte 1(O autor)

```

55
56     public void prepareAdicionar(ActionEvent actionEvent){
57         concurso = new Concurso();
58         financiador = new Financiador();
59     }
60     public void adicionar(ActionEvent actionEvent){
61         dao = new ConcursoDao();
62         concurso.setFinanciador(financiador);
63         dao.merge(concurso);
64         FacesContext.getCurrentInstance().addMessage(null,
65             new FacesMessage(FacesMessage.SEVERITY_INFO, "Operação feita com sucesso","",""));
66     }
67     public void prepareAlterar(ActionEvent actionEvent){
68         concurso = (Concurso)listDM.getRowData();
69         financiador=new Financiador();
70         financiador.setIdFinanciador(concurso.getFinanciador().getIdFinanciador());
71     }
72     public void excluir(ActionEvent actionEvent){
73         concurso = (Concurso)listDM.getRowData();
74         dao = new ConcursoDao();
75         dao.delete(concurso);
76         FacesContext.getCurrentInstance().addMessage(null,
77             new FacesMessage(FacesMessage.SEVERITY_INFO, "Excluído com sucesso","",""));
78     }
79     public DataModel getListar(){
80         dao = new ConcursoDao();
81         List<Concurso> lista = dao.listAll(Concurso.class);
82         listDM = new ListDataModel(lista);
83         return listDM;
84     }

```

Figura 13: Classe ConcursoBean - parte 2 (O autor)

É através dos métodos getters/setters que a página acessará os atributos, por isso precisa dos getters/setters para seus atributos.

```

90  public void setConcurso(Concurso concurso) {
91      this.concurso = concurso;
92  }
93
94  public ConcursoDao getDao() {
95      return dac;
96  }
97
98  public void setDao(ConcursoDao dao) {
99      this.dao = dao;
100 }
101
102 public DataModel getListDM() {
103     return listDM;
104 }
105
106 public void setListDM(DataModel listDM) {
107     this.listDM = listDM;
108 }
109
110 public List<Concurso> getFilteredConcursos() {
111     return filteredConcursos;
112 }
113
114 public void setFilteredConcursos(List<Concurso> filteredConcursos) {
115     this.filteredConcursos = filteredConcursos;
116 }
117

```

Figura 13: Classe ConcursoBean-parte 3 (O autor)

Ver o apêndice 1, onde encontramos a codificação do sistema e as ferramentas de concepção.

3.2.3.2. Descrição do Sistema: Características e Funcionalidades do Sistema desenvolvido (implementação e testes)

O sistema em causa possibilitará a Gestão da Informação na DIPLAC-CEE. Contudo é importante destacar o controlo de acesso da informação, feito através da autenticação e da autorização dos usuários do sistema através da tela inicial de acesso do sistema representada pela Figura 14.

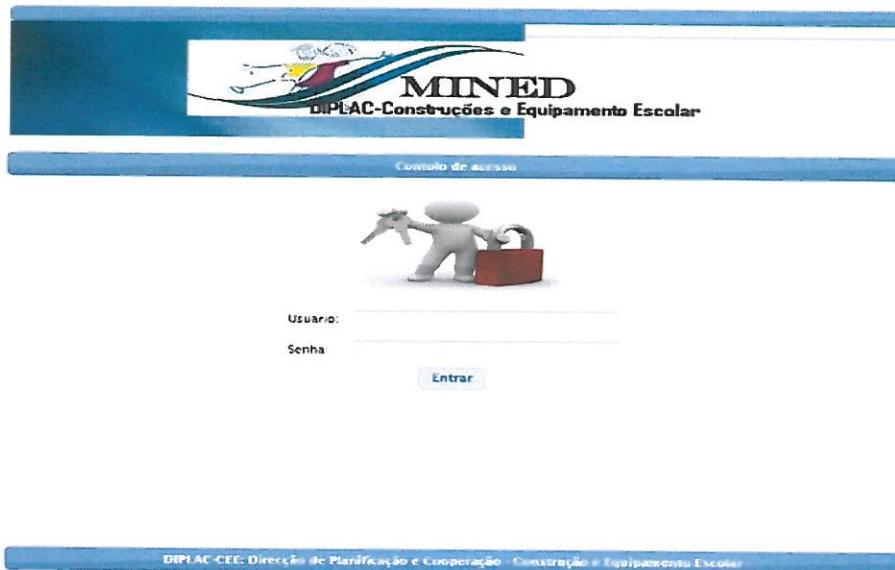


Figura14: Tela Inicial do Sistema (O autor)

O Sistema busca a informação na base de dados e verifica o usuário e a senha e logo em seguida após o processo de autenticação vem a autorização que normalmente significa *quem acessa o que?*

Após a autenticação, se o usuário e a senha forem correcto o sistema o direcionará para a página principal do sistema, como ilustra a figura 15.

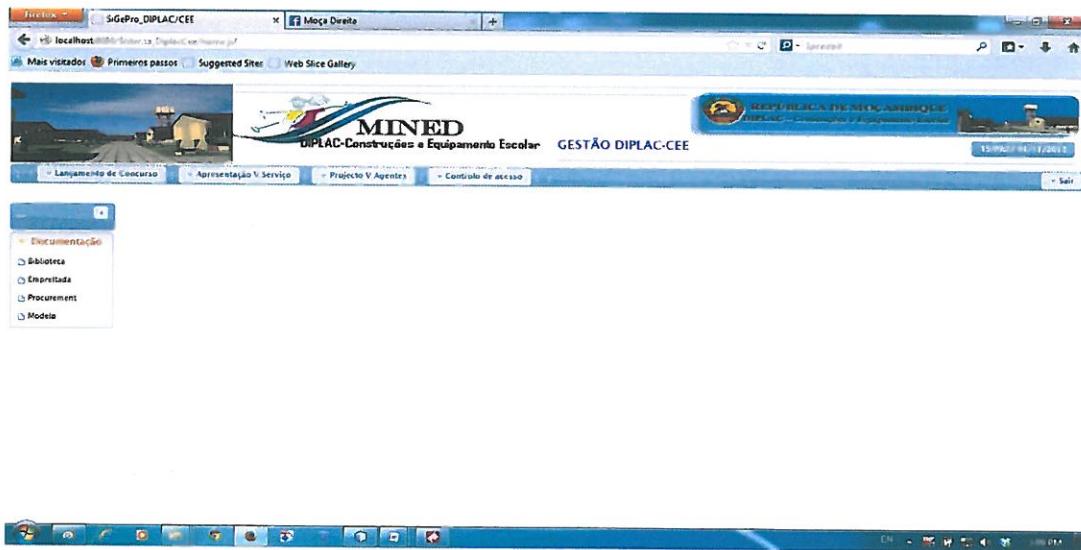


Figura 15: Tela principal (O autor)

O processo de Inserção, Actualização, Exclusão, Consulta e Impressão agora já é informatizado. É feito indo ao Menu pretendido (neste caso Lançamento de concurso →

Concurso), conforme mostra a figura 16.

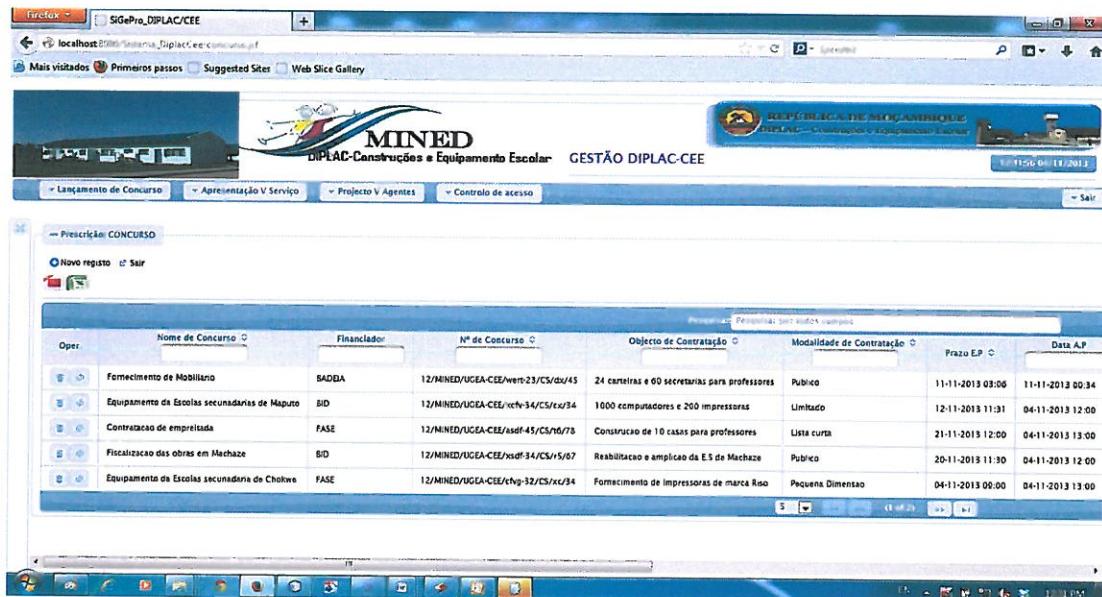


Figura 16: Formulário de Concurso (O autor)

Para a Inserção é só clicar no botão “Novo registo”, que aparecerá o formulário como mostra a figura 17. No formulário como podemos ver, temos o botão “Confirmar” para gravar o registo, botão “Cancelar” para limpar o registo, botão “Sair” para o fecho do formulário caso não queira fazer nenhum registo.

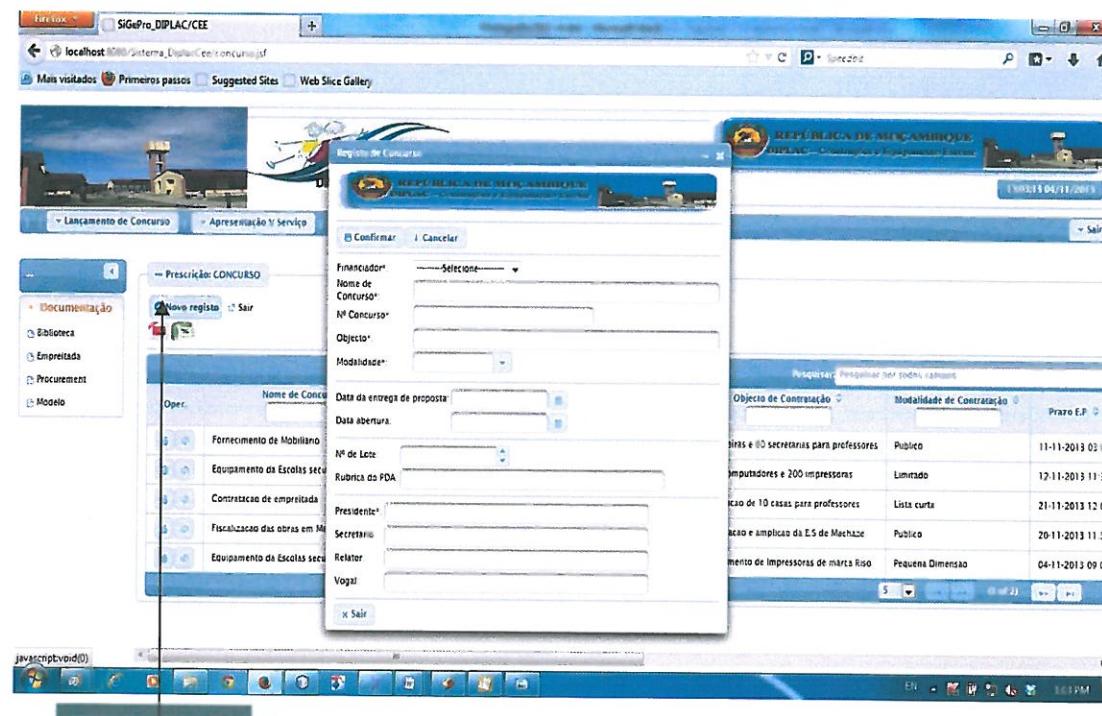


Figura 17: Formulário de Inserção (O autor)

Para a actualização é só clicar no botão “Actualizar”, que se encontra no lado direito da tabela na coluna “Oper.”, que aparecerá o formulário como mostra a figura 18. No formulário como podemos ver, temos o botão “Confirmar” para actualizar, botão “Cancelar” para limpar o registo, botão “Sair” para o fecho do formulário caso não queira fazer nenhuma actualização. Salientar que o botão para “Excluir: o registo encontra-se ao lado do botão “Actualizar” como ilustra a figura abaixo.

Figura 18: Formulário de Actualização (O autor)

Para Exportar os dados é só clicar no Icon do Pdf ou Excel, conforme o desejar, que aparecerá o a impressão como mostra a figura 19 (Export por pdf).

Nome de Concurso	Financiador	N de Concurso	Prazo E.P.	Data A.P
Fornecimento de Mobiliario	BADEIA	I2/MINED/UG EA-CEE/wert-23/CS/dx/45	11-11-2013 03:06	11-11-2013 00:34
Equipamento da Escolas secundarias de Maputo	BID	I2/MINED/UG EA-CEE/xcfv-34/CS/cx/34	12-11-2013 11:31	04-11-2013 12:00
Contratacao de empreitada	FASE	I2/MINED/UG EA-CEE/asdf-45/CS/t6/78	21-11-2013 12:00	04-11-2013 13:00
Fiscalizacao das obras em Machaze	BID	I2/MINED/UG EA-CEE/xsdf-34/CS/r5/67	20-11-2013 11:30	04-11-2013 12:00
Equipamento da Escolas secundaria de Chokwe	FASE	I2/MINED/UG EA-CEE/cfg-32/CS/xc/34	04-11-2013 09:00	04-11-2013 13:00
Contratacao de 2 Tecnicos Arquitectos	FASE	I2/MINED/UG EA-CEE/dfgh-43/CS/cv/45	04-11-2013 14:30	04-11-2013 15:00

Figura 19: Exportação por Pdf (O autor)

Para consultar os Manuais de Gestão de Empreitada basta clicar no menu Documentação →Biblioteca (ou outro menu que o desejar), como mostra a figura 20.

The screenshot shows a web-based document management system. The top navigation bar includes links for 'Lançamento de Concurso', 'Apresentação V Serviço', 'Projeto V Agentes', and 'Controlo de acesso'. The main content area is titled 'Biblioteca' and displays a list of documents categorized under 'Biblioteca 1/3', 'Biblioteca 2/3', and 'Biblioteca 3/3'. Each category contains several documents with checkboxes next to their names. The sidebar on the left also lists 'Empreitada', 'Procurement', and 'Modelo' under 'Documentação'.

Categoria	Documento	Ação
Biblioteca 1/3	LEI 8/2002 de 12 de Fevereiro que cria o Sistema de Administração Financeira do Estado	<input type="checkbox"/>
	DECRETO 15/2010, de 24 de Maio que aprova o Regulamento de Administração de Operações de Obras Públicas e Fornecimento de Bens e Serviços do Estado	<input type="checkbox"/>
	DM 147/2006, de 08 de Setembro que aprova os Documentos de Concurso para Bens e Serviços	<input type="checkbox"/>
	IM 153 Elaboração de Projetos de Edifícios-Actividades técnicas	<input type="checkbox"/>
	Resumo dos Requisitos e Modalidades relativamente ao valor estimado da contratação	<input type="checkbox"/>
	TBTF Fiscal de Obras	<input type="checkbox"/>
	LEI 6/2004, de 17 de Junho que prova os mecanismos complementares de combate à corrupção	<input type="checkbox"/>
	DECRETO 45/2004, de 29 de Setembro que aprova o Regulamento de Aplicação do Impacto Ambiental	<input type="checkbox"/>
	DM 146/2009, de 24 de Junho que aprova o Regulamento-Tipo da Organização do SDR	<input type="checkbox"/>
	IM 152 elaboração de projetos de Edifícios-Arquitectura	<input type="checkbox"/>
Biblioteca 2/3	Decreto 42/2008, de 04 de Novembro que aprova os Documentos de Concurso para a Empreitada de Obras Públicas	<input type="checkbox"/>
	DECRETO 23/2004, de 20 de Outubro que aprova o Regulamento do Sistema de Administração Financeira do Estado	<input type="checkbox"/>
	DM 145/2006, de 08 de Setembro que aprova os Documentos de Concurso para a Empreitada de Obras Públicas	<input type="checkbox"/>
	IM 221 Manual de Operação, Uso e Manutenção	<input type="checkbox"/>
	Manual de Procedimentos para a contratação de Obras Públicas, fornecimento de Bens e Serviços de Empreitada de Serviços ao Estado	<input type="checkbox"/>
	Ficha de Informação Ambiental Preliminar	<input type="checkbox"/>
	Instrumento de Apoio a escolha da Modalidade de Contratação	<input type="checkbox"/>
	Programa de Gestão da Empreitada	<input type="checkbox"/>
	Resumo dos Requisitos e Modalidades relativamente ao valor estimado da contratação	<input type="checkbox"/>
	Resumo dos Requisitos e Modalidades relativamente ao valor estimado da contratação	<input type="checkbox"/>
Biblioteca 3/3	Decreto 42/2008, de 04 de Novembro que aprova os Documentos de Concurso para a Empreitada de Obras Públicas	<input type="checkbox"/>
	Decreto 45/2004, de 29 de Setembro que aprova o Regulamento de Aplicação do Impacto Ambiental	<input type="checkbox"/>
	Decreto 23/2004, de 20 de Outubro que aprova o Regulamento do Sistema de Administração Financeira do Estado	<input type="checkbox"/>

Figura 20: Formulário de Manuais Gestão de Empreitada (O autor)

Na tabela para processo de consulta, os campos encontram-se junto do cabeçalho da coluna da tabela, como mostra a figura 21.

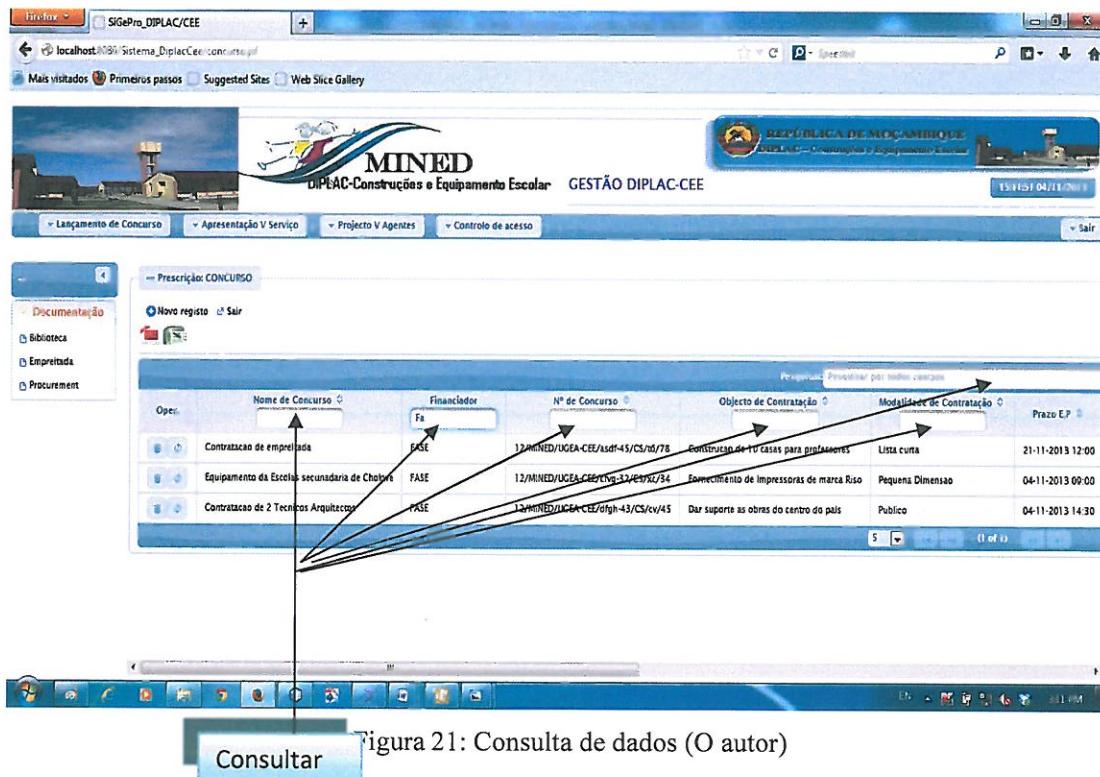


Figura 21: Consulta de dados (O autor)

3.2.4. Fase de Transição

Esta fase abrange a entrega do *software* ao cliente e a fase de testes. O objectivo desta fase é disponibilizar o sistema, tornando-o disponível e compreendido pelo usuário final. O relatório de *feedback* do usuário assim como dos clientes, o manual do utilizador e o Testes Beta do Sistema (*Outputs*), são os resultados esperados desta fase.

O relatório de *feedback*, objectiva trazer uma apreciação e o ponto de vista do usuário final do sistema, servindo como uma ferramenta para medir a satisfação, no que diz respeito aos resultados objectivados no processo da recolha dos requisitos. Ver anexo1.

O Manual do utilizador objectiva guiar o utilizador pelas várias funcionalidades disponíveis no Sistema de Gestão de Projectos de Construção da DIPLAC-CEE. Ver o apêndice 1.

O Teste Beta do Sistema (*Outputs*) objectiva colocar em acção as operações automatizadas de modo a descobrir defeitos e deficiências das operações do sistema. No que diz respeito aos *Outputs* do Sistema em causa, fez-se duas apresentações. Na primeira fez-se a apresentação do projecto a ser concebido e na segunda, foram colocados em prática, isto é, automatizados conforme mostram as figura 14, 15, 16, 17, 18, 19, 21.

CAPITULO IV - CONCLUSÃO

Este capítulo conclui o trabalho, indicando trabalhos futuros, além de expor as considerações finais em relação aos objectivos pretendidos e a bibliografia consultada.

4.1. Futuras implementações

Uma outra funcionalidade será futuramente implementada no sistema, é a disponibilização na internet, o que oferecerá o acesso em qualquer lugar deste momento que esteja ligado a internet e esteja cadastrado no sistema.

4.2. Consideração final

Devido ao problema acima supracitado na gestão da informação na DIPLAC-CEE, surgiu a necessidade de assim responder a esse problema concebendo um Sistema de Gestão de Informação. Para a sua concepção usou-se RUP para a modelagem e Java para a sua construção. Desta feita convém registar algumas conclusões:

Ao final deste trabalho pode-se considerar que ocorreu o cumprimento do escopo delimitado, sendo que tanto a empresa quanto o desenvolvedor do sistema conseguiram identificar as necessidades dos usuários, isso fez com que este projecto conseguisse chegar a sua fiel conclusão.

No que diz respeito ao foco deste trabalho que é a automatização dos processos de inserção, actualização, exclusão e consulta de dados, foi totalmente finalizada o que constitui agrado por parte dos funcionários. E espera-se que com a automatização dos processos haja melhoraria na gestão da Informação e integridade da informação em paralelo com o crescimento da instituição.

O sistema desenvolvido foi realizado utilizando algumas tecnologias e metodologias vistas ao longo do curso e que foram consideradas de grande importância para que o projecto tenha sido concebido, para tanto, foram realizadas pesquisas bibliográficas visando alcançar o maior conhecimento possível a respeito das tecnologias e metodologias necessárias para a sua concepção.

4.3. Recomendações

Para garantir uma boa gestão e controlo do sistema é importante que haja formação aos utilizadores do sistema, para que não haja resistência a mudança, visto que passarão a usar o sistema automatizado. Salientar que é exigido no mínimo aos utilizadores conhecimentos de informática na óptica de utilizador.

Recomenda-se que a implementação seja imediata, para melhorar o sistema de acordo com o feedback dos funcionários e garantir que futuras implementações sejam desenvolvidas.

Em cada acção realizada no sistema é necessário garantir que a operação foi realizada com sucesso de modo a garantir integridade dos dados. É recomendado também que se atente aos erros gerados pelo sistema e de imediato seja contactado o gestor do sistema. Recomenda-se por último que após a implementação se faça o uso para garantir sejam visíveis os resultados esperados.

4.4. Referência Bibliográfica

A Importância dos SI nas Organizações. Disponivel em: <<http://www.si-organizacoes.blogspot.com>> . Acessado em : 12 de Setembro de 2013

Apostila_adm-sistema-de-informação. Disponível em: <<http://www.win2pdf.com>>. Acesso: 12 de Setembro de 2013

BOOCH, Grady; RUMBAUGH, James; JACOBSON, Ivar. UML: guia do usuário. Trad. Fabio Freitas. Rio de Janeiro: Campus, 2000.

Comparação entre Java e algumas Linguagens de Programação. Disponível em: <http://pt.wikipedia.org/wiki/Compara%C3%A7%C3%A3o_entre_linguagens_de_programa%C3%A7%C3%A3o> . Acessado em: 08 de Setembro de 2013.

Desenvolvimento e Manutenção de Sistemas. Disponivel em: <<http://www.teccconcursos.com.br/conteudo/questoes/19725?materia=ti---engenharia-de-software&banca=cespe>>. Acessado em: 08 de Setembro de 2013.

JAVASERVER Faces Technology. Disponivel em: <<http://java.sun.com/javaserverfaces/index.jsp>>, acesso em 13 de Setembro de 2013.

LAUDON, Kenneth C; LAUDON, Jane P. **Sistemas de informação com internet**. Rio de Janeiro: LTC, 1999, 389 p.

MANUAL de Referencia do MySQL. Disponivel em: <dev.mysql.com/doc/mysql/mysql_pt/Índex.html>, acesso em: 12 de Setembro de 2013

ORT, Edward. **Introducing the Java EE 6 Platform**. Disponível: <<http://java.sun.com/features>>, acesso: 13 de Setembro de 2013.

ORT, Edward. **The Basics of GlassFish**. Disponível em: <<http://java.sun.com/features/>>, acesso: 20 de Setembro de 2013.

PRESSMAN, ROGER S., *Engenharia de Software- (6^a edição)*, São Paulo, Ed. McGrawHill, 2006, 371p.

PRIMEFACES. Primefaces quickstart tutorial. Disponivel em: <<http://java.dzone.com/articles/Primefaces-quickstart-tutorial>>, acesso em 10 de Setembro de 2013

SANTOS, Rui Rossi dos. **Programando em Java 2 - Teoria & Aplicações**. Rio de Janeiro: Axcel Books, 2004.

ANEXO

Anexo 1: Relatório de Feedback

Vista a apresentação do Sistema de Gestão de Projectos na DIPLAC-CEE, que objectiva automatizar os processos envolvidos na gestão da informação, solicitamos ao estimado funcionário a responder o seguinte questionário.

Exponha a sua opinião assinalando na alternativa correta

1. Repartição

- Procurement
 Contabilidade
 Construções
 Informática.

2. Tem alguma apreciação no sistema apresentado? (Se a resposta for não passe para a pergunta a ultima questão)?

- Sim. Não.

3. Acha que o sistema vai trazer alguma melhoria no que concerne a:

Inserção dos dados?

- Mau.
 Razável.
 Bom.

3.2. Consulta de informação?

- Mau.
 Razável.
 Bom.

3.3. Flexibilidade na disponibilidade da informação?

- Mau.
 Razável.
 Bom.

3.4. Integridade da informação?

- Mau.
 Razável.
 Bom.

3. Comentários

Obrigado pela colaboração

Vista a apresentação do Sistema de Gestão de Projectos na DIPLAC-CEI, que objectiva automatizar os processos envolvidos na gestão da informação, solicitamos ao estimado funcionário a responder o seguinte questionário.

Exponha a sua opinião assimilando na alternativa correta

1. Repartição

- Procurement
 Contabilidade
 Construções.
 Informática.

2. Tem alguma apreciação no sistema apresentado? (Se a resposta for não passe para a pergunta a ultima questão)?

- Sim. Não.

3. Acha que o sistema vai trazer alguma melhoria no que concerne a:

Inserção dos dados?

- Mau.
 Razoável.
 Bom.

3.2. Consulta de informação?

- Mau.
 Razoável.
 Bom.

3.3. Flexibilidade na disponibilidade da informação?

- Mau.
 Razoável.
 Bom.

3.4. Integridade da informação?

- Mau.
 Razoável.
 Bom.

3. Comentários

o sistema será absolutamente positivo se for adequado nos momentos necessários do processo de precativo e assim como nos procedimentos
Obrigado pela colaboração meus, de modo a tornar-se informação
e gestor.

Atentamente
28/11/2013

Vista a apresentação do Sistema de Gestão de Projectos na DIPLAC-CEE, que objectiva automatizar os processos envolvidos na gestão da informação, solicitamos ao estimado funcionário a responder o seguinte questionário.

Exponha a sua opinião assinalando na alternativa correta

1. Repartição

- Procurement
- Contabilidade
- Construções.
- Informática.

2. Tem alguma apreciação no sistema apresentado? (Se a resposta for não passe para a pergunta a última questão)

- Sim.
- Não.

3. Acha que o sistema vai trazer alguma melhoria no que concerne a:

Inserção dos dados?

- Mau.
- Razoável.
- Bem.

3.2. Consulta de informação?

- Mau.
- Razoável.
- Bem.

3.3. Flexibilidade na disponibilidade da informação?

- Mau.
- Razoável.
- Bem.

3.4. Integridade da informação?

- Mau.
- Razoável.
- Bem.

3. Comentários

*Precisa ser mais adaptado à organização
ao projeto, em uso na instituição.*

Obrigado pela colaboração

Vista a apresentação do Sistema de Gestão de Projectos na DIPLAC-CFE, que objectiva automatizar os processos envolvidos na gestão da informação, solicitamos ao estimado funcionário a responder o seguinte questionário.

Exponha a sua opinião assinalando na alternativa correta

1. Repartição

- Procurement
 Contabilidade
 Construções.
 Informática.

2. Tem alguma apreciação no sistema apresentado? (Se a resposta for não passe para a pergunta a ultima questão)?

- Sim. Não.

3. Acha que o sistema vai trazer alguma melhoria no que concerne a:

Inserção dos dados?

- Mau.
 Razoável.
 Bom.

3.2. Consulta de informação?

- Mau.
 Razoável.
 Bom.

3.3. Flexibilidade na disponibilidade da informação?

- Mau.
 Razoável.
 Bom.

3.4. Integridade da informação?

- Mau.
 Razoável.
 Bom.

3. Comentários

Obrigado pela colaboração

Anexo 2- Roteiro de entrevista

<u>ROTEIRO DE PERGUNTAS</u>	
1.	A quanto tempo é funcionário desta Instituição?
2.	Tem algum conhecimento sobre: a) O que base de dados e para que serve? b) O que é um Sistema? c) O que Sistema da Gestão de Informação e para que serve?
3.	Nesta instituição usa-se: a) Alguma base de Dados b) Sistema da Gestão de Informação? _____
4.	A quanto tempo trabalham assim? _____
5.	O que acha sobre o Sistema usado nesta instituição. a) Tem algum benefício? Comenta na sua resposta. b) Tem alguma vantagem? Comenta na sua resposta _____
6.	Como melhorar o Sistema existente de modo a: a) Trazer ou aumentar os benefícios existentes? b) A sanar ou minimizar os constrangimentos existentes? c) Quem seria beneficiado? d) Até que ponto facilitaria nos processos envolvidos na gestão actual? Obrigado pela colaboração

Apêndice 1 -DVD: Codificação do Sistema ferramentas de concepção e Manual de utilizador