****

**Universidade Federal de Pernambuco**

**Centro de Informática**

**Ciências da Computação**

**Trabalho de Graduação**

**Adição de colaboração e realização de validação heurística qualitativa da ferramenta de consultas XML – ViXQueL**

**Aluno:** Eduardo José de Vasconcelos Matos ([*ejvm@cin.ufpe.br*](mailto:ejvm@cin.ufpe.br))

**Orientador:** Prof. Dr. Fernando da Fonseca de Souza *(*[*fdfd@cin.ufpe.br*](mailto:fdfd@cin.ufpe.br))

**Co-orientador:** Petrus Rangel Bastos ([*petrusbastos@gmail.com*](mailto:petrusbastos@gmail.com))

Recife, Julho de 2010

**Assinaturas**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Fernando da Fonseca de Souza (Orientador)**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Eduardo José de Vasconcelos Matos (Aluno)**

“O sol já nasceu na estrada nova e mesmo que eu impeça, ele vai brilhar.  
Lembra se puder, se não der esqueça – de algum jeito vai passar.”

**Oswaldo Montenegro**

# Resumo

XML tornou-se padrão quando se trata de troca de informações entre diversas aplicações. Deste modo, sistemas de gerenciamento de bancos de dados XML nativos têm se tornado peça importante na infra-estrutura de empresas que lidam com essa diversidade de fontes de dados.

As interfaces gráficas disponíveis para a recuperação destas informações não demonstram preocupação com o modo ou conhecimento necessário para desenvolver tal atividade. Com essas ferramentas, é necessário ter conhecimento de uma linguagem textual de consulta aos SGBD em questão. Com a ViXQueL, tornou-se possível que essas informações encontradas nos SGBD XML nativos fossem recuperadas de forma mais intuitiva sem a necessidade de uma linguagem complexa como a XQuery. Entretanto, caso fosse o desejo de criar consultas visuais de forma colaborativa ou de um auxílio de uma segunda pessoa para criar consultas mais elaboradas, isso ainda não seria possível.

Para resolver essa limitação, este trabalho visa estender ViXQueL,uma ferramenta de criação de consultas visuais para bancos de dados XML nativos, para permitir colaboração entre desenvolvedores. Para tanto, será mostrada uma revisão da literatura sobre ferramentas que apresentam propostas semelhantes a da ViXQueL. Posteriormente, será mostrada a ViXQueL e sua arquitetura. Em seguida, será destacada a importância da colaboração na atualidade, o seu desenvolvimento na história, e como algumas funcionalidades voltadas para a colaboração foram implementadas e adicionadas à ViXQueL. Finalmente, será realizada uma avaliação heurística sobre a usabilidade e atratividade da linguagem.

**Palavras-chave:** Colaboração, Validação heurística, XQuery, SGBD XML nativo, Sedna, MDA.

# Abstract

XML has become standard when it is the case of information exchange between different applications. Thus, native XML databases have become important piece in companies’ infrastructure that deal with such a variety of data sources.

Graphical interfaces available for retrieving such information usually are not concerned with the knowledge needed to perform such an activity. To use such tools, one must have knowledge of a textual query language to access underlying DBMSs. ViXQueL made it possible that information found in native XML DBMSs could be recovered more intuitively with no need to use a complex language such as XQuery. However, to create visual queries collaboratively or to get assistance from other person to create more elaborate queries, it still could not be possible.

To face such a limitation, this work is aimed at extending ViXQueL, an authoring tool for visual queries to native XML databases, to allow collaboration between developers. To achieve this, it is presented a review of the literature on similar tools that have purposes similar to ViXQueL. After, it is described ViXQueL and its architecture. Following, it is highlighted the importance of collaboration in actuality, its development in history, and how some features geared to collaboration were implemented and added to ViXQueL. Finally, it is presented a heuristic evaluation on the usability and attractiveness of the language.

**Key words**: Collaboration, Validation heuristics, XQuery, Native XML DBMS Sedna, MDA

# Agradecimentos

Nesse trecho mais íntimo do trabalho, gostaria de agradecer a tudo e a todos que fazem parte de mim. É difícil não deixar passar ninguém, pois se fosse realizar minha vontade, o agradecimento daria um número infinitamente maior do que as poucas páginas que aqui vão se seguir.

Gostaria de agradecer primeiramente ao Deus dentro de nós e que é parte daquilo que somos. Agradeço também à família que recentemente se despediu de um integrante memorável, meu Tio Beto, mas que está cada vez mais forte e unida. Meus pais Cláudio e Magaly, minhas irmãs Nara (e meu cunhado Yuri) e Rebeca e minha sobrinha Laís. Foram eles minha companhia e meus alicerces desde o nascimento até o ponto que me encontro nesta longa caminhada.

Agradeço a minha incondicional namorada e companheira Raquel que esteve comigo em todas as dificuldades desse trabalho e espero que em tantos outros futuros. Cuidou de mim e, apesar de todas as dificuldades, me deu força e amor pra seguir em frente.

Agradeço ao meu orientador Fernando Fonseca, pelo cuidado e dedicação com que me acompanhou, desde o início, na trajetória desta etapa tão importante para a minha vida. Além destes, gostaria de agradecer a todos os professores da Graduação em Ciências da Computação do Centro de Informática da Universidade Federal de Pernambuco que de alguma forma participaram desta caminhada.

Agradeço com muita alegria a meu co-orientador e amigo Petrus Bastos (e sua esposa Viviann) por ter me guiado e amparado nos momentos em que me perdi e desesperei. E também, por ter criado a ViXQueL, e junto a Bill (Rafael Carício) ter dado continuidade nesse projeto ao meu lado dias, noites, madrugadas inclusive em fins de semanas.

Agradeço, por fim, aos amigos dos projetos da minha vida: do Marista Recife (em especial a Lula e Marina), do CEFET-PE (em especial a Adriana e Bárbara), do Especial (em especial às Ninjets e Eli), do Atletismo de Pernambuco e da Motorola (à Flavinha, aos meus velhos, aos bolsistas e chefes por terem apostado em mim em mais um grande projeto da minha vida).

A todos um muito obrigado!

Sem vocês esta conquista não seria possível!

Índice

[1. Introdução 1](#_Toc266450037)

[1.1. Motivação e Objetivo 1](#_Toc266450038)

[1.2. Estrutura do trabalho 2](#_Toc266450039)

[2. Trabalhos Relacionados 4](#_Toc266450040)

[2.1. XQBE 5](#_Toc266450041)

[2.2. GXQL 6](#_Toc266450042)

[2.3. BBQ 8](#_Toc266450043)

[2.4. XQueryViz 9](#_Toc266450044)

[2.5. XGI 10](#_Toc266450045)

[2.6. Considerações Finais 11](#_Toc266450046)

[3. A Ferramenta ViXQueL 13](#_Toc266450047)

[3.1. Funcionalidade 13](#_Toc266450048)

[3.2. Arquitetura 14](#_Toc266450049)

[3.3. O sistema 15](#_Toc266450050)

[3.4. A geração do código XQuery 18](#_Toc266450051)

[3.4.1. Metamodelos 19](#_Toc266450052)

[3.4.2. Transformações entre modelos 23](#_Toc266450053)

[3.5. Expressividade da ViXQueL 26](#_Toc266450054)

[3.6. Considerações Finais 27](#_Toc266450055)

[4. Colaboração 28](#_Toc266450056)

[4.1. História 28](#_Toc266450057)

[4.2. Tipos de Colaboração (tempo x espaço) 30](#_Toc266450058)

[4.3. Arquitetura da Colaboração 32](#_Toc266450059)

[4.4. Funções Adicionadas 34](#_Toc266450060)

[4.4.1. Novas Telas da ViXQueL 35](#_Toc266450061)

[4.4.1.1. Chat 37](#_Toc266450062)

[4.4.1.2. Salvar/Recuperar consultas 37](#_Toc266450063)

[4.4.1.3. Visualização da criação da consulta em tempo real 38](#_Toc266450064)

[4.5. Considerações Finais 39](#_Toc266450065)

[5. Validação 40](#_Toc266450066)

[5.1. Interface gráfica 40](#_Toc266450067)

[5.2. Modelo de validação de interfaces web por usuários 42](#_Toc266450068)

[5.3. Itens da Validação de ViXQueL 45](#_Toc266450069)

[5.4. Validação da Ferramenta ViXQueL 47](#_Toc266450070)

[5.5. Resultados Obtidos 49](#_Toc266450071)

[6. Trabalhos Futuros e Considerações Finais 52](#_Toc266450072)

[6.1. Considerações Finais 52](#_Toc266450073)

[6.2. Trabalhos Futuros 53](#_Toc266450074)

[Referências 55](#_Toc266450075)

# Lista de Figuras

[Figura 2‑1 – Interface gráfica da ferramenta XQBE 6](#_Toc266450076)

[Figura 2‑2 – Interface gráfica da ferramenta GXQL 7](#_Toc266450077)

[Figura 2‑3 – Interface gráfica da ferramenta BBQ 8](#_Toc266450078)

[Figura 2‑4 – Interface gráfica da ferramenta XQueryViz 10](#_Toc266450079)

[Figura 2‑5 – Interface gráfica da ferramenta XGI 11](#_Toc266450080)

[Figura 3‑1 – Arquitetura da ferramenta ViXQueL 14](#_Toc266450081)

[Figura 3‑2 – Tela de gerenciamento de conectividade da ViXQueL 16](#_Toc266450082)

[Figura 3‑3 – Tela de gerenciamento de bases de dados 17](#_Toc266450083)

[Figura 3‑4 – Tela principal da ferramenta ViXQueL. 17](#_Toc266450084)

[Figura 3‑5 – Diagrama de estrutura do metamodelo ViXQueL 19](#_Toc266450085)

[Figura 3‑6 – Exemplo de um modelo da ViXQueL 21](#_Toc266450086)

[Figura 3‑7 – Exemplo de um modelo da XQuery. 23](#_Toc266450087)

[Figura 3‑8 – Processo de transformação da ViXQueL para código XQuery 25](#_Toc266450088)

[Figura 4‑1 – Tipos de colaboração: tempo x espaço – traduzida de (45). 31](#_Toc266450089)

[Figura 4‑2 – Arquitetura da colaboração. 33](#_Toc266450090)

[Figura 4‑3 – Tela inicial da ViXQueL 35](#_Toc266450091)

[Figura 4‑4 – Escolha/Criação de sala. 36](#_Toc266450092)

[Figura 4‑5 – Nova tela de consulta da ViXQueL 36](#_Toc266450093)

[Figura 4‑6 – Menu de ações: Salvar, Abrir, Limpar tela, (des)habilitar chat, 38](#_Toc266450094)

[Figura 4‑7 – Botões de *Lock/Unlock, respectivamente.* 38](#_Toc266450095)

[Figura 5‑1 – Pintura rupestre, papiro, incunábulo, livro e hieróglifo. 40](#_Toc266450096)

[Figura 5‑2 – Padrão de atração e de usabilidade durante os estágios de interação com websites – traduzida de (51). 44](#_Toc266450097)

[Figura 5‑3 – Instruções iniciais da avaliação. 49](#_Toc266450098)

# Lista de Tabelas

[Tabela 5‑1 – Média das notas da validação 50](#_Toc266367396)

# Lista de Quadro

[Quadro 2‑1 – Resumo das características avaliadas dos trabalhos relacionados. 12](#_Toc266450099)

[Quadro 3‑1 – Funções das áreas da Ferramenta ViXQueL 18](#_Toc266450100)

[Quadro 3‑2 – Funcionalidades da XQuery suportadas pela ViXQueL. 26](#_Toc266450101)

# Introdução

Este capítulo relata as principais motivações para a realização deste trabalho, lista os objetivos de pesquisa almejados, além de mostrar a estrutura de cada capítulo restante da presente dissertação.

## Motivação e Objetivo

O mundo corporativo atual está em constante mudança e tais mudanças são atribuídas às decisões tomadas pelos gestores a partir de um leque de informações. Para que essas informações sejam acessadas de forma rápida, eficiente e, sempre que possível, remotamente, os gerentes necessitam de ferramentas que disponibilizem esse serviço.

No entanto, para que essas decisões sejam tomadas de forma eficiente, normalmente ocorrem em um conjunto composto por mais de um gestor. No mundo cada vez mais globalizado, nem sempre é possível a presença física desses responsáveis pelas decisões num lugar em comum. Desta maneira, não há nada melhor para o gestor do que a possibilidade dele mesmo recuperar a informação desejada em conjunto com outros gestores de forma colaborativa, sem depender de nenhum especialista da área de informática.

Outra questão importante que deve ser levada em consideração é a utilização de uma ferramenta de consulta a banco de dados XML nativo. Esta necessidade é gerada a partir da diversidade de fontes e formatos de dados necessários para a tomada de decisão em uma empresa. Assim, o uso de XML (1) e dos SGDB XML nativos (2) vieram para suprir essa problemática. Entretanto, para consultar esses dados em formato XML, é requerido o conhecimento numa linguagem não-trivial: a XQuery (3). E, para resolver esse problema, a ViXQueL (4) foi criada, permitindo a criação de consultas visuais (gráficas) que abstraem a necessidade do conhecimento da linguagem XQuery.

Em contrapartida, a ViXQueL ainda não permitia a colaboração entre seus usuários, bem como sua interface não havia sido validada. Essa validação também é de suma importância por apresentar diretivas para um constante melhoramento da ferramenta. É mostrado também o que pode estimular o usuário a utilizar ViXQueL e o que pode ser melhorado com relação à intuitividade da ViXQueL.

Para resolver essas últimas problemáticas, este trabalho tem por objetivo geral apresentar uma solução ao problema citado anteriormente. Assim, apresenta-se significativas adições de funcionalidades à ViXQueL, uma ferramenta visual para facilitar o acesso a SGBD XML nativos. De modo geral, a ferramenta se propõe a facilitar o acesso de seus usuários a informações contidas em bases de dados XML de forma colaborativa. Através do uso de um ambiente *Web* visual, as modificações propostas permitem que o usuário acesse tais informações de forma colaborativa mesmo que não tenha nenhum conhecimento de tecnologias relacionadas a XML.

Adicionalmente, foi verificada a necessidade de uma validação utilizando métodos formais sobre a usabilidade e atratividade de ViXQueL. Com essa validação, busca-se alcançar a confirmação da premissa de que qualquer usuário não-especialista em XQuery é capaz de realizar consultas a bancos de dados XML nativos.

Com o intuito de atingir os objetivos gerais deste trabalho, foi necessário definir alguns objetivos específicos:

* Adicionar funcionalidades que permitam colaboração entre mais de um usuário, tais como:
  + Um chat para possibilitar a comunicação dentro da ViXQueL, sem a necessidade de o usuário ficar alternando telas a fim de discutir a construção da consulta;
  + A funcionalidade de salvar uma consulta construída por um ou mais usuários e recarregá-la caso desejado; e
  + Permitir a visualização da consulta ao mesmo tempo em que ela está sendo criada.
* Validar a usabilidade e interface da ViXQueL utilizando técnica heurística com usuários não-especialistas em XQuery ou qualquer linguagem de consulta a banco de dados.

## Estrutura do trabalho

Além deste capítulo introdutório, no qual foram apresentadas a motivação e os objetivos da ferramenta proposta, esta dissertação é composta por mais cinco capítulos, como segue.

O Capítulo 2, Trabalhos Relacionados, trata das abordagens relacionadas, fornecendo uma visão geral das principais ferramentas existentes. Inicialmente, são descritas as principais abordagens referenciadas na literatura que compartilham a mesma preocupação deste trabalho, e que fornecem estratégias diferentes para o tratamento do problema. Por fim, realiza uma análise destes trabalhos e aponta limitações em comum.

O Capítulo 3, A Ferramenta ViXQueL, aborda a primeira versão da ferramenta. O capítulo se inicia com uma visão geral do que é a ferramenta e como ela foi concebida e, posteriormente, são explanadas as principais funcionalidades da mesma. Por fim, são apresentados a arquitetura da ferramenta e os detalhes de sua implementação.

O Capítulo 4, Colaboração, mostra como a colaboração foi e continua sendo importante. Em seguida, são apresentados os tipos de colaboração em relação ao seu tempo e espaço. Por fim, são detalhadas as características colaborativas adicionadas à ViXQueL.

O Capítulo 5, Validação, apresenta o desenvolvimento da comunicação visual e a importância do design. Em seguida, ratifica o valor da validação de interfaces gráficas e suas conseqüências. Posteriormente, apresenta a metodologia utilizada para a validação da ViXQueL e os resultados obtidos através dessa validação.

O Capítulo 6, Trabalhos futuros e considerações finais, apresenta as conclusões do trabalho com um breve resumo dos resultados obtidos, dificuldades encontradas e principais contribuições, além de apontar as limitações e propostas de trabalhos futuros.

# Trabalhos Relacionados

Neste capítulo, são abordados os trabalhos relacionados e destacadas as deficiências comuns entre eles. O capítulo se inicia com um breve histórico do surgimento de linguagens gráficas de consultas. Posteriormente, destaca os trabalhos relacionados, e por fim, apresenta uma análise das deficiências comuns entre os trabalhos relacionados mais relevantes.

O interesse por linguagens gráficas e de consultas já existe há algum tempo. A linguagem G, no ano de 1987, foi apresentada por Cruz et al.(5). Esta surgiu como uma das primeiras linguages de consulta gráfica orientada a objetos. No ano seguinte, Cruz et. al. (6) apresentaram a linguagem sucessora G+. Com isso, deu-se início ao surgimento de diversas outras linguagens gráficas e de consulta, dentre as quais pode-se citar Graphlog (7) e Good (8). Esta última passou a oferecer uma notação uniforme para sistemas de banco de dados orientados a objetos, onde os nós representavam os objetos e a representação dos relacionamentos se dava através de pontes. Alguns anos depois, em 1995, surgiu uma linguagem que tinha uma notação parecida com a linguagem Good, linguagem esta chamada de G-Log(9). Essa é uma linguagem gráfica baseada em lógica que torna possível a representação e a consulta de objetos complexos por meio de grafos rotulados direcionados.

Em 1998, despontam as linguagens com o interesse em consultar dados semi-estruturados. A primeira a surgir com este intuito foi a WG-Log(10). Essa linguagem foi construída para consultar páginas da internet e dados semi-estruturados, adicionando à G-Log algumas funcionalidades relacionadas à hipermídia. Em 2001, surge a XML-GL (11; 12), a primeira linguagem descendente da WG-Log. A XML-GL foi a primeira linguagem de consulta visual para XML, contudo, foi projetada bem antes da XQuery(3).

Após analisar todo este histórico, é possível notar a importância das linguagens gráficas de consulta. Ao longo desse capítulo, serão destacadas as ferramentas que utilizam linguagens gráficas de consulta baseadas no uso da XQuery e suas principais funcionalidades. Além disto, será feita uma avaliação das ferramentas utilizando-se de alguns critérios importantes os quais também motivaram o desenvolvimento deste trabalho. Por fim, serão apontadas as principais limitações destas ferramentas.

Antes de apresentar de fato as abordagens relacionadas, é importante frisar quais foram os critérios utilizados para a avaliação das ferramentas relacionadas. Tais critérios foram:

* O nível de conhecimento requerido do usuário em XQuery e tecnologias relacionadas a XML;
* Se a ferramenta requer aprendizado de uma linguagem visual por parte dos usuários;
* Se houve a realização de validação com usuários leigos em XQuery e tecnologias relacionadas a XML;
* Se a ferramenta foi projetada para funcionar na web;
* Se a ferramenta realiza consultas em um SGBD XML nativo;
* Se a ferramenta permite a colaboração entre usuários na criação de consultas;
* Se uma validação da ferramenta sobre sua usabilidade e/ou atratividade foi realizada.

## XQBE

Braga et. al. (13) conceberam a XQuery By Example (XQBE). Esta é baseada no paradigma Query By Example (QBE)(14) e, por conseguinte, demonstrou que uma linguagem visual é efetiva quando se trata de dar suporte a expressões de consulta intuitivas. A XQBE é destacada como primeira linguagem de consulta gráfica. Esta simultaneamente permitia, aos usuários não conhecedores de linguagens complexas de consulta, acessar e modificar fontes de dados XML. A sua efetividade é ainda maior quando: as construções básicas, o paradigma de consulta envolvido e a abstração visual são compatíveis com a linguagem e o modelo de dados da camada inferior. Por esse motivo, enquanto a QBE é a linguagem de consulta relacional, baseada no uso de tabelas, a XQBE é baseada no uso de árvores, assim aderindo ao modelo de dados hierárquico de XML.

O projeto da ferramenta XQBE tem como principal objetivo prover facilidade de uso. Houve, também, uma preocupação em criar uma linguagem de consulta o mais expressiva possível tornando-a diretamente mapeável para a XQuery. Deste modo, qualquer interface gráfica capaz de executar no topo de qualquer implementação existente da XQuery. Contudo, para representações visuais de transformações complexas, a linguagem XQBE tende de forma inerente a tornar-se de difícil compreensão, devido ao crescimento maciço do número de nós. E, por esse motivo, a linguagem é fortemente desencorajada, funcionando melhor de maneira efetiva apenas com transformações simples.

Em relação a sua implementação, a XQBE tem uma arquitetura cliente-servidor e funciona da seguinte maneira: o cliente possui apenas o editor visual de consultas para o usuário; e o servidor processa todas as transformações e, se requisitado, executa a consulta utilizando um mecanismo de execução de XQuery. A comunicação é realizada entre o cliente e o servidor através de uma representação interna das consultas em um formato intermediário que é basicamente uma descrição de grafos XQBE em XML. A interface gráfica da ferramenta pode ser vista na .

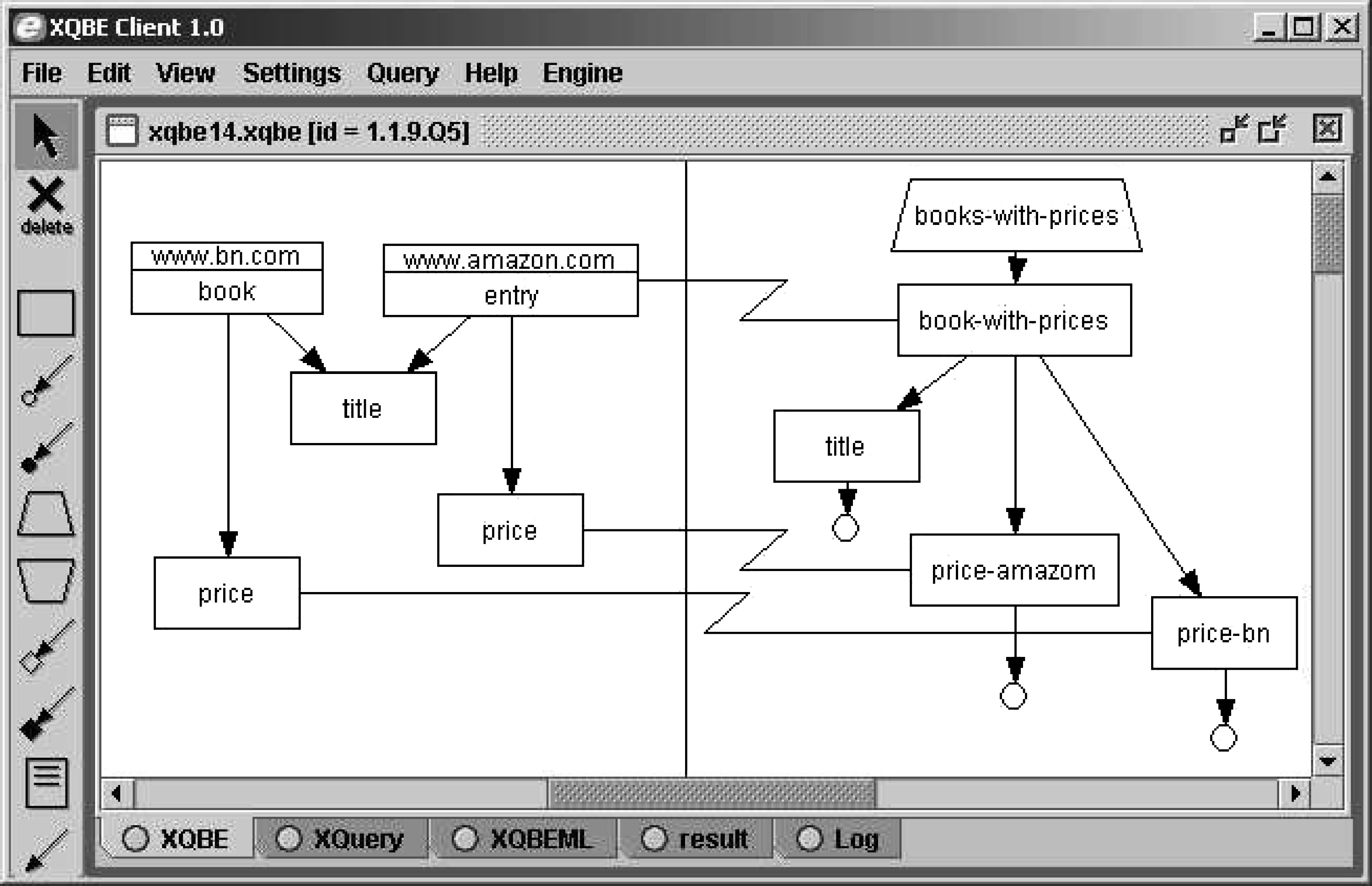


Figura ‑ – Interface gráfica da ferramenta XQBE

Apesar da motivação de Braga et. al. em criar uma ferramenta amigável para usuários leigos, o uso de diversos elementos similares para a representação de diferentes conceitos e a necessidade de um conhecimento básico em XQuery prejudicaram a XQBE em atingir completamente sua meta inicial.

## GXQL

A ferramenta Graphical XQuery Language (GXQL) foi criada para sanar as problemáticas encontradas pela XQBE por Z. Qin et. al. (15). A proposta deles era mais expressiva do que a XQBE: as consultas de alta complexidade na XQBE eram representadas de maneira mais clara e fácil na GXQL. Bem como, algumas consultas que a XQBE não podia representar seriam, na GXQL, bem expressadas.

Diferentemente da XQuery By Exemple, a Graphical XQuery Language tinha uma abordagem baseada em janelas, bordas e cores, como saída para a problemática da usabilidade e discernimento para as pessoas não familiarizadas com a linguagem XQuery. A ferramenta de Qin utiliza janelas alinhadas para representação dos relacionamentos entre entidades descendentes. Além disto, os elementos e atributos de uma determinada entidade são diferenciados graficamente. O uso das janelas aninhadas possibilita a visualização hierárquica das consultas elaboradas e, também, o uso de funcionalidades como auxílio ao usuário na construção de consultas, para que este não necessite fornecer todas as entradas textualmente, ou mesmo, criar todas as consultas do início. A Figura 2‑2 apresenta a interface gráfica da ferramenta.

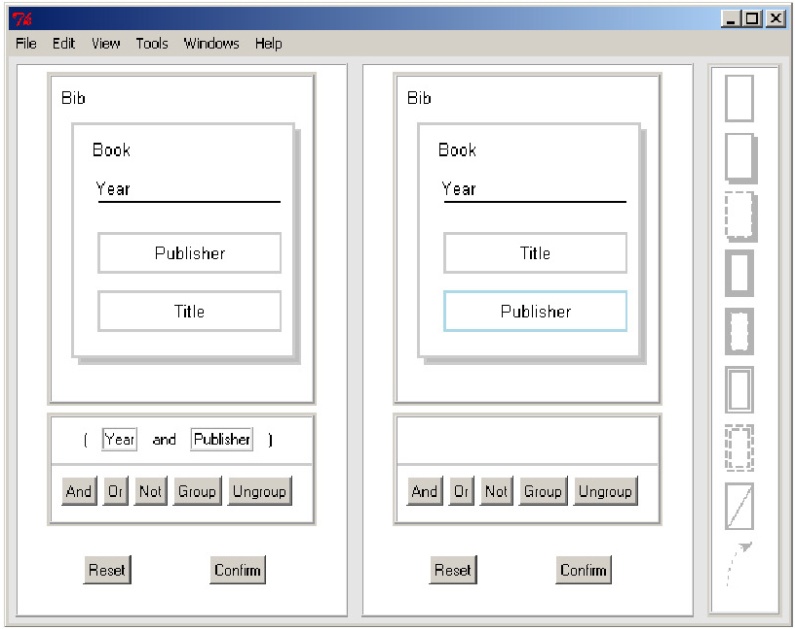


Figura ‑ – Interface gráfica da ferramenta GXQL

Apesar de todo esforço despendido na GXQL visando maior usabilidade e uma curva de aprendizagem relativa à abordagem da XQBE, pode-se citar como falhas da Graphical XQuery Language: a necessidade de conhecer todos os significados das bordas dos retângulos para a construção da consulta; o fato de que os usuários freqüentemente perdem a visão geral da construção gráfica da XQuery devido às novas telas que são abertas durante o processo de criação da consulta XQuery; e a necessidade de barras de rolagem relativas à grande quantidade de informação gráfica mostrada.

Por fim, apesar de os autores afirmarem que a ferramenta gráfica proposta é amigável para os usuários finais, esta não foi submetida a nenhum tipo de validação com usuários. Assim, não é possível confirmar que a ferramenta de fato auxilia os usuários sem experiência na construção de consultas.

## BBQ

A ferramenta BBQ (*Blended Browsing and Querying*) apresentada por Munroe et. al. (16) é baseada na estrutura de navegação de diretórios do sistema operacional Windows (17), o qual permite operações de drill-down para facilitar a navegação.

A BBQ é um fragmento do projeto MIX (*Mediation of Information using XML*) (18). Como parte cliente do projeto, a ferramenta busca facilitar consultas e navegações em uma ou mais fontes de dados XML, o refinamento de consultas iterativamente, e ainda, a descoberta de estruturas dos dados. As fontes de dados são apresentadas em um formato de interface multi-documentos, onde cada fonte é associada a uma janela contendo seus respectivos dados e esquema mostrados lado a lado no quadro. Tanto os esquemas quanto os dados são apresentados em estrutura de árvore, como se fossem diretórios, nos quais os usuários podem navegar e adicionar condições. A interface gráfica da ferramenta pode ser vista na Figura 2‑3.

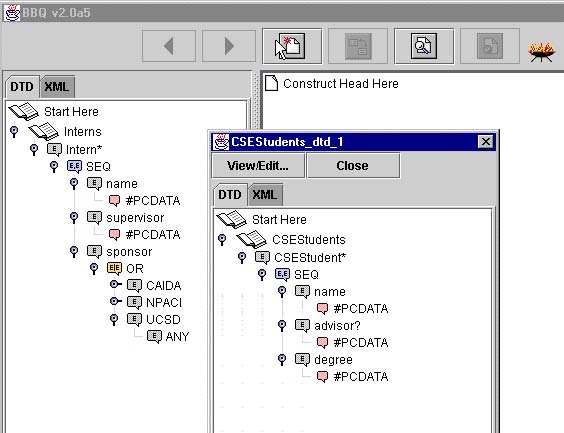


Figura ‑ – Interface gráfica da ferramenta BBQ

Para a construção das consultas, a ferramenta BBQ utiliza dois paradigmas de pesquisa: o de bancos de dados relacionais e o da web. No paradigma utilizado pelos bancos de dados relacionais, as consultas na BBQ são orientadas a esquema e utilizam DTD(19). Já quando a pesquisa está voltada para web, assume-se que o usuário não tem certeza absoluta dos resultados esperados da consulta. E assim, o sistema enfatiza a navegação de esquemas e dados e os resultados da consulta podem ser refinados iterativamente.

Esse refinamento é possível porque os resultados de uma consulta podem ser re-utilizados como fontes em futuras consultas. Os usuários têm a possibilidade de construir um documento de resultado da consulta, e este se torna fonte de dados de primeira classe dentro da BBQ, significando que o documento pode ser navegado, consultado, ou ainda, utilizado para construir outro documento de resultado de consulta semelhante a si.

Mesmo dando simultaneamente suporte a mais de uma fonte de dados, a BBQ faz uso de uma DTD adicionada ao mecanismo de inferência que ajuda a refinar os processos de consultas XQuery. A ferramenta também é uma das pioneiras no sentido de focar na interação com o usuário, bem como possibilitar a construção de consultas XQuery de forma mais rápida e efetiva. Entretanto, a BBQ tem como público-alvo usuários já familiarizados com a estrutura aninhada dos dados XML e com alguns conceitos de linguagens de consulta a bancos de dados, tais como, junções e agregações. E, por esse motivo, Munroe et. al. (16) sugerem *webmasters*, desenvolvedores ou administradores de bancos de dados como usuários para a sua ferramenta.

## XQueryViz

A ferramenta XQueryViz (20) foi apresentada por Karam et. al. com o propósito de auxiliar os usuários finais com a criação de consultas XQuery / XPath (21). Isto se tornou possível com a manipulação direta da representação em árvore do XML Schema (22) usado em sua abordagem gráfica.

A XQueryViz é baseada no uso de ícones e ligações entre esses ícones. Tais ligações são criadas utilizando enlaces de dados. E, devido à existência destes enlaces, o usuário consegue visualizar como os dados estão sendo passados nas construções visuais. Além disto, a ferramenta é uma das que representa graficamente o maior número de construções da XQuery, visto que esse era um dos objetivos principais do seu projeto. Apesar de ter sido moldada para ajudar no aprendizado de estudantes da linguagem XQuery, a XQueryViz é tão complexa quanto a linguagem XQuery em si. A Figura 2‑4 apresenta a tela da ferramenta.

Por essa razão, e por não terem sido realizados testes com usuários que não detêm conhecimento em XQuery/XPath, não se pode afirmar que a ferramenta realmente alcança seus objetivos primários. Assim, apesar de ser, atualmente, a ferramenta com maior representatividade da linguagem XQuery, não é razoável pensar que a ferramenta possa ser utilizada em todos os seus recursos sem que o usuário possua um mínimo conhecimento de uma linguagem tão complexa quanto XQuery/XPath.

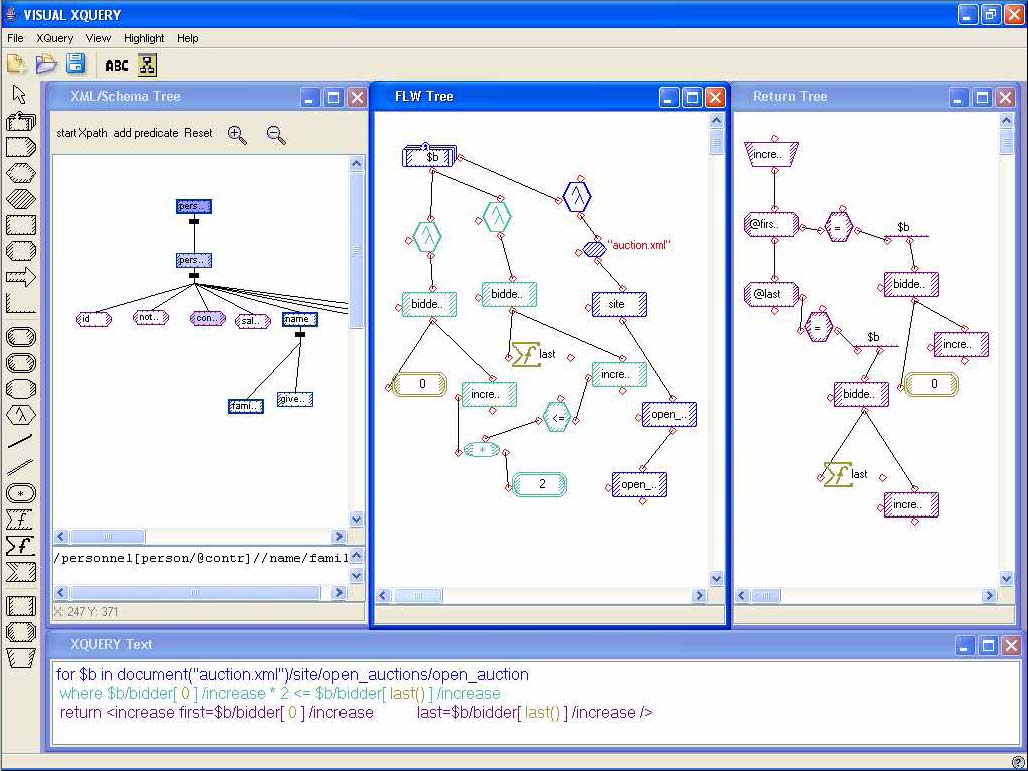


Figura ‑ – Interface gráfica da ferramenta XQueryViz

## XGI

A XQuery Graphical Interface (XGI) (23), apresentada por Li et. al.,é a proposta mais próxima de uma ferramenta que permite a criação de consultas XQuery por usuários não-especialistas através de uma ferramenta intuitiva.

A ferramenta permite a realização de construções de consultas de forma intermediária entre os dados não-estruturados e estruturados. E, na tentativa de orientar os usuários sobre o esquema existente, a XGI exibe uma árvore fonte navegável e permite a manipulação gráfica dos elementos dessa estrutura.

A XQuery Graphical Interface é uma ferramenta de arquitetura WEB, visto que foi criada inicialmente para pesquisadores biomédicos e que os mesmos não deveriam ter a responsabilidade de instalar e manter as atualizações do *software*. Além disto, o ambiente WEB permite que qualquer pesquisador possa construir suas consultas de qualquer lugar. A interface da XGI é mostrada a seguir na Figura 2‑5:

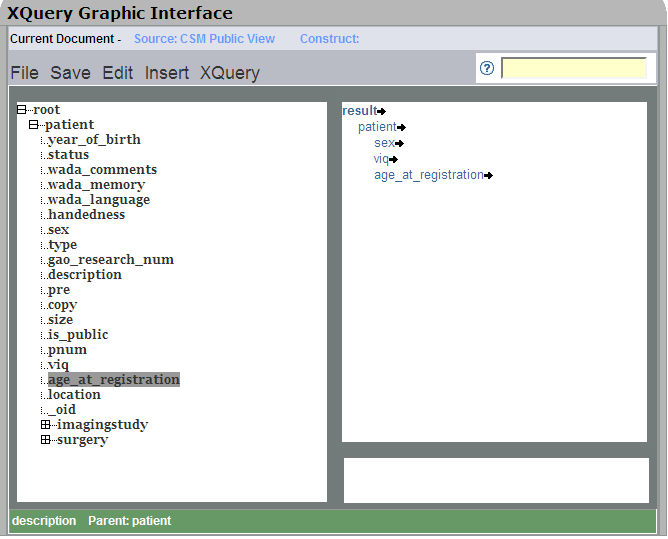


Figura ‑ – Interface gráfica da ferramenta XGI

Apesar da proposta inicial de Li ter sido permitir que usuários não-especialistas em XQuery pudessem criar consultas de modo mais fácil, essa proposta não foi totalmente satisfatória. E isso ocorreu pelo fato de que alguns conceitos de XML devem ser conhecidos para que o usuário opere a ferramenta. Pode-se citar como exemplo dessa necessidade, o conhecimento de elementos como o ROOT para que se possa iniciar a construção das consultas, bem como o conceito de árvore usado em XML para que seja criada a estrutura de retorno desta mesma consulta.

Embora tenha havido validação de usabilidade, esta foi realizada apenas com um usuário especialista. E, portanto, com essa avaliação, os autores não são capazes de afirmar que a ferramenta realmente auxilia os não-conhecedores das tecnologias relacionadas a XML na construção de consultas. Além disso, há, por parte dos próprios criadores da XGI, uma aceitação de que a proposta de facilitar a criação de consultas pela popularização da XQuery não foi totalmente alcançada, pois a ferramenta adiciona somente valor a usuários experientes em linguagens complexas como a XQuery.

## Considerações Finais

Inicialmente, deve-se destacar o fato de que nenhuma das ferramentas citadas anteriormente permite consultas a sistemas de gerenciamento de bancos de dados XML nativos. Logo, não permitem aos usuários gozarem das funcionalidades presentes nos SGBD como, por exemplo: recuperação de falhas, controle de concorrência, segurança, entre outros.

Também, sobre as ferramentas estudadas, não foi possível identificar em nenhuma delas, a metodologia aplicada para a geração de consultas XQuery a partir das ferramentas gráficas. A XBQE e a XGI apenas citam a utilização de uma BNF (24), embora não explicitem as técnicas e o funcionamento do processo de transformação.

E, apesar das diversas ferramentas e suas linguagens visuais correspondentes vistas até este ponto assegurarem que ajudam o trabalho de recuperação de informação por parte de usuários não-conhecedores da XQuery, nenhuma delas realizou testes de validação com usuários detentores deste nível de conhecimento, o que poderia comprovar suas afirmações.

Por fim, é notório que apesar do conhecido potencial de uma ferramenta WEB, apenas uma das ferramentas analisadas foi desenvolvida para este ambiente. Assim, há uma significante perda no potencial de colaboração entre os usuários da maioria destas ferramentas. No Quadro 2‑1 a seguir, pode-se ver um resumo das principais características das linguagens visuais analisadas e suas ferramentas:

Quadro ‑ – Resumo das características avaliadas dos trabalhos relacionados.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Ferramenta | Nível de conhecimento requerido do usuário | Requer aprendizado de linguagem visual | Validação com usuários leigos em XQuery | Funciona na Web | Consulta SGBD XML Nativo | Permite colaboração |
| XQBE | Básico | Sim | Não realizado | Não | Não | Não |
| BBQ | Básico | Não | Não realizado | Não | Não | Não |
| GXQL | Nenhum | Sim | Não realizado | Não | Não | Não |
| XQueryViz | Avançado | Sim | Não realizado | Não | Não | Não |
| XGI | Básico | Não | Não realizado | Sim | Não | Não |
| ViXQueL\* | Nenhum | Não | ***Avaliação Preliminar*** | Sim | Sim | Não |

* A ferramenta VIXQueL, alvo do trabalho, será apresentada em detalhes no capítulo seguinte.

# A Ferramenta ViXQueL

O objetivo deste capítulo é introduzir a concepção do protótipo da solução sugerida, destacando aqui a sua arquitetura, implementação e funcionalidades.

## Funcionalidade

Um dos intuitos deste trabalho é adicionar funcionalidades colaborativas a ViXQueL (4), uma ferramenta de criação de consulta visual a sistemas de gerenciamento de bancos de dados XML nativo. Não menos importante, o segundo objetivo do trabalho propõe a realização de uma validação de ViXQueL por parte do usuário em relação a sua usabilidade e atratividade. Portanto, com esses objetivos, torna-se imprescindível apresentar uma descrição de ViXQueL, seu conjunto de funções, suas características, sua arquitetura e suas limitações.

A ferramenta ViXQueL pode ser dividida em 3 (três) partes principais. Estas partes estão relacionadas ao gerenciamento do servidor de banco de dados, ao gerenciamento das bases e à criação das consultas visuais.

Na primeira parte da ferramenta, gerenciamento do servidor de banco de dados, existem algumas ações indispensáveis para permitir a realização do mesmo. Iniciar e parar o servidor de banco de dados são as principais ações para que o protótipo possa funcionar corretamente. Além disso, é importante fornecer uma maneira de avaliar se o servidor de banco de dados está iniciado ou não, permitindo a verificação de seu funcionamento.

Com relação à segunda parte, gerenciamento das bases de dados, existem outras ações fundamentais para a realização do gerenciamento. Criar, iniciar, parar e remover bases de dados são as suas ações básicas. Outra importante ação é a inserção de documentos XML nas bases de dados. Para o funcionamento adequado de todas essas ações, exceto criar, é necessária a seleção prévia de uma base de dados.

Todas as funcionalidades da primeira e segunda parte da ferramenta, relacionadas ao gerenciamento do servidor de banco de dados, bem como ao gerenciamento das bases de dados, são auxiliares e visam dar suporte à principal funcionalidade do sistema que é a criação visual de consultas. Esta funcionalidade permite que os usuários não-especialistas realizem consultas em bases de dados XML nativas.

A criação visual de consultas permite que os usuários criem consultas na Web de forma visual e intuitiva, visualizem o retorno das consultas, e ainda caso desejem, visualizem a consulta XQuery gerada a partir da sua representação visual. E, um dos fatores principais na usabilidade da ferramenta, é a abstração de vários conceitos da XML durante o processo de construção das consultas.

## Arquitetura

Primeiramente, antes de abordar as funcionalidades da ferramenta ViXQueL, é interessante ressaltar a sua estrutura proposta. Esta estrutura se compõe da arquitetura da ferramenta, tecnologias utilizadas, além da maneira como elas se relacionam. A arquitetura da ferramenta ViXQueL é apresentada na Figura 3‑1:

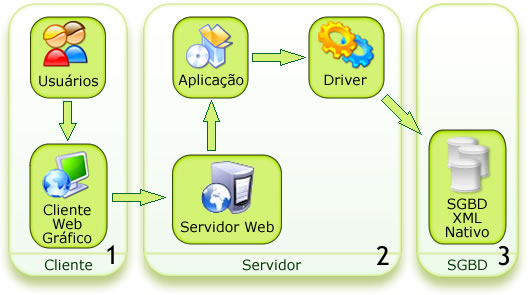


Figura ‑ – Arquitetura da ferramenta ViXQueL

O sistema é baseado na arquitetura cliente-servidor e, para facilitar a compreensão, é dividido em 3 blocos principais: o 1º é composto por um cliente web gráfico para a construção das consultas, ligamento e desligamento do servidor e escolha dos arquivos ou bases XML para serem carregados. O 2º é composto pela aplicação residente na parte servidor da arquitetura, o servidor Web Apache (Tomcat) (25). Esta aplicação é escrita na linguagem Java e utiliza um driver que permite a comunicação entre a aplicação do 1º bloco e do 3º bloco. Este último composto pelo SGBD XML nativo.

O cliente web gráfico no bloco 1 é o que permite a criação de consultas desejadas pelo usuário apenas arrastando alguns componentes, sem haver a necessidade de aprendizado de uma linguagem visual completa. Este cliente foi codificado em *Javascript*(26), *HTML* (*HyperText* *Markup* *Language*) (27), *CSS* (*Cascading Style Sheets*) (28) e *Scriptaculous*(29), além de utilizar o framework *PGF* (*Prototype Graphic Framework*) (30) para gerar as entidades visuais exibidas no navegador. E, apesar do *PGF* ser um framework bem conceituado, foi necessário também, desenvolver e ajustar algumas funcionalidades, com o intuito de atender às necessidades da ferramenta durante o processo de desenvolvimento.

Um dos grandes desafios encontrados no desenvolvimento deste cliente web gráfico foi a geração, exibição e manipulação das entidades gráficas em tempo de execução. Para isso, foi realizada uma extensa pesquisa e avaliação de diversos frameworks existentes para a realização destas tarefas de modo satisfatório.

O framework Jscanvas (31) foi utilizado inicialmente para a criação da interface, mas durante o desenvolvimento, surgiu a necessidade da integração de várias telas da ferramenta. A partir deste ponto, o Jscanvas foi substituído pelo framework PGF, que atendeu de forma adequada às necessidades do desenvolvimento.

Para a criação das consultas, as informações necessárias para a construção e transformação da suas representações visuais em representações textuais, são armazenadas no próprio cliente em variáveis existentes no código Javascript.

No segundo bloco, a aplicação residente no Tomcat é a parte de ViXQueL que realiza a transformação da consulta visual para a consulta textual, além de ser responsável por conectar-se ao SGBD XML nativo e executar as consultas construídas. A recuperação do esquema XML é outra responsabilidade importante deste bloco. Tal esquema é utilizado para a navegação das entidades e para construção das consultas. Este esquema é extraído através da verificação das entidades e atributos existentes na base e, por isso, não tem ligação nenhuma com DTD (19) ou XML Schema (22).

O SGBD XML nativo Sedna(32) está situado no bloco 3 e compõe o lado da arquitetura que é responsável por armazenar os documentos XML utilizados nas consultas construídas. Todos os documentos XML referenciados são alocados em uma *collection* a fim de permitir a consulta a mais de um documento XML. E, também, buscando uma melhor usabilidade, tanto os elementos, quanto os atributos são representados da mesma forma na estrutura em forma de árvore para a consulta XQuery visual.

## O sistema

Nesta seção, é descrito o processo completo de conexão do cliente gráfico até o SGBD XML nativo através do servidor web. Também é apresentada a funcionalidade de criação de novas bases, bem como é fornecida uma apresentação detalhada de todos os objetos encontrados na interface gráfica da aplicação do lado cliente, a qual é utilizada na criação de consultas que são realizadas na base de dados.

A Figura 3‑2 mostra a interface da ViXQueL responsável pela conexão com o SGBD XML nativo e pelas operações de gerenciamento do servidor de banco de dados. Estas operações podem ser: iniciar e parar o servidor, verificar se o servidor está ou não iniciado, gerenciar bases de dados, bem como seus dados. Além disto, esta área de gerenciamento tem o item “Query information”, que levará para a tela de criação de consultas.

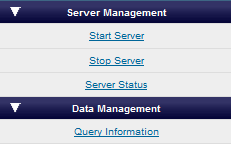


Figura ‑ – Tela de gerenciamento de conectividade da ViXQueL

Janelas pop-up são mostradas durante as operações de gerenciamento, bem como em toda a aplicação, com os resultados das consultas implementadas, mensagens de erro na conexão e fechamento da conexão. Nestas operações, o cliente se comunica com o SGBD XML nativo utilizando a tecnologia Asynchronous JavaScript And XML (AJAX)(33).

A mostra a tela que contém as operações de gerenciamento das bases de dados. Esta permite a escolha da base de dados a ser utilizada nas operações de gerenciamento de informações e consultas. Nesta tela, pode-se ainda iniciar, parar e excluir bases de dados selecionadas, além de permitir a criação de novas bases de dados.

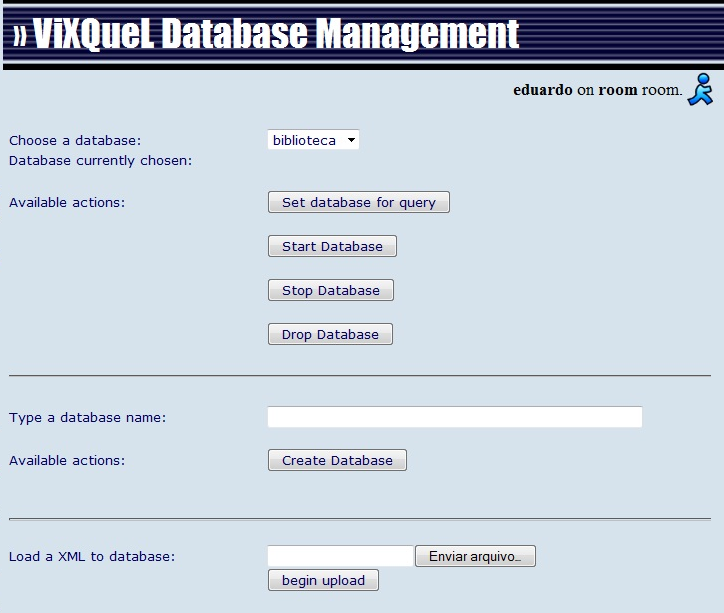


Figura ‑ – Tela de gerenciamento de bases de dados

A tela de criação de consultas é dividida em oito partes principais (Figura 3‑4). Cada uma das funcionalidades existentes nesta tela é apresentada e explicada logo abaixo e está numerada de 1 a 8, como pode ser visto no Quadro 3‑1.

## D:\TG\Docs\imagens\tela_de_criacao_de_consultas.jpg

Figura ‑ – Tela principal da ferramenta ViXQueL.

Quadro ‑ – Funções das áreas da Ferramenta ViXQueL

|  |  |  |
| --- | --- | --- |
| Área | Nome | Função |
| 1 | Árvore de metadados | É a representação gráfica dos metadados dos documentos XML armazenados na base de dados escolhida. São entidades *drag and drop* que devem ser colocadas na área de criação de consulta. |
| 2 | Área de criação de consulta | Maior área na tela que possibilita que items *drag and drop* sejam posicionados a fim de criar uma consulta. |
| 3 | Elementos de restrições de relacionamento entre entidades | São as representações gráficas das restrições permitidas entre duas entidades. Estas podem ser: igual, diferente, maior que, menor que, maior ou igual que, e menor ou igual que. |
| 4 | Elementos de restrições de ordenação e retorno da consulta | São as representações gráficas das restrições de ordenação e retorno da consulta. Estas podem indicar quais entidades devem ser retornadas pela consulta e qual seu tipo de ordenação. |
| 5 | Consulta XQuery gerada | É a consulta XQuery gerada através da estruturação realizada na área (2). A presença desta visualização na tela de criação de consultas é opcional e pode ser configurada pelo usuário na área (7). |
| 6 | Resultado da consulta | É o resultado da consulta gerada. Após o término da estruturação da consulta, esta pode ser executada através do botão encontrado na área (7). |
| 7 | Botões de ação | São os botões que permitem executar uma consulta ou configurar a ViXQueL. |
| 8 | Área de remoção | Se houver algum erro na escolha de uma entidade, esta deve ser arrastada para a área de remoção para ser excluída da área (2). Neste caso, todas as restrições pertencentes ou ligadas à entidade excluída também serão excluídas. |

## A geração do código XQuery

Para a criação das consultas a um SGBD XML nativo de forma mais simplificada e tangível àqueles que não dominam linguagens complexas, como, por exemplo, a XQuery, é necessário a criação de uma linguagem de modelagem específica de domínio: a ViXQueL. Tal linguagem permite o uso de conceitos e regras que são parte da problemática específica que está sendo endereçada. Através da etapa de formalização deste modelo em um metamodelo, é possível construir novas transformações que levam a modelos mais concretos. Analogamente, um modelo de consulta XQuery, após submetidos a novas transformações, transformam-se em uma consulta em código XQuery.

A geração do código XQuery é baseada nos conceitos de **Model-driven architecture** (MDA) e as tecnologias **Meta-Object Facility** (MOF) (34) da OMG no Eclipse Modeling Framework (EMF)(35). O EMF é um framework de modelagem e geração de código do Eclipse(36) para construção de ferramentas e aplicações baseadas em modelos de dados estruturados. Tais modelos implementam um subconjunto do núcleo padrão do MOF que é adequado para o trabalho com metamodelagem(35).

Este subconjunto do MOF no EMF é chamado de Ecore e estipula as regras utilizadas na ViXQueL para a geração de consultas XQuery. Para a realização destas transformações, dois metamodelos são utilizados. O primeiro metamodelo é o ViXQueL, o qual define o conjunto de estrutura e elementos possíveis para a criação de consultas. O segundo é o metamodelo que representa as construções da linguagem XQuery usadas pela ViXQueL para a criação de consultas. Este último metamodelo é representativo o bastante para permitir a geração do código XQuery em formato textual, que por conseguinte, permite a realização de consultas em um Sistema de Gerenciamento de Bases de Dados XML nativo.

## Metamodelos

O metamodelo da ViXQueL, que tem seu diagrama apresentado na Figura 3‑5, tem como elemento principal a entidade ViXQueL. Este elemento representa uma consulta e, através do seu atributo *database*, especifica a base de dados à qual esta consulta deve ser aplicada. A entidade ViXQueL contém também um elemento *rootNode*, que representa o nó principal do documento XML e é uma instância da entidade *DatabaseNode*.

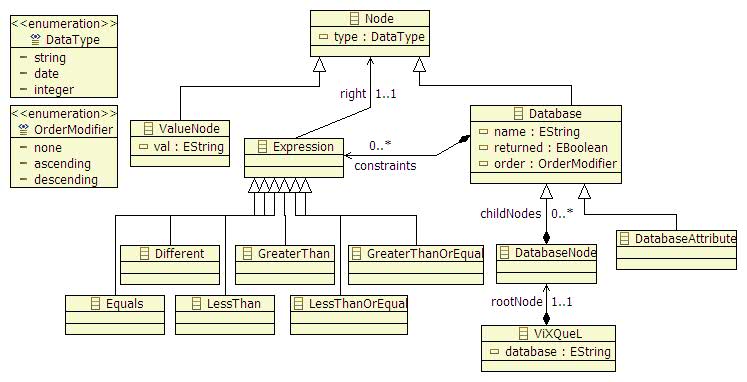


Figura ‑ – Diagrama de estrutura do metamodelo ViXQueL

Uma entidade *DatabaseNode* pode conter zero ou mais elementos *childNodes*. Estes elementos *childNodes* são instâncias da entidade *Database*, que, por sua vez, podem ser instâncias das entidades *DatabaseNode* ou *DatabaseAttribute*. A entidade *Database* representa os elementos de XML utilizados na consulta e contém três atributos que determinam seu papel na consulta, são estes: Um nome, um indicador que informa se é um dos elementos retornados pela consulta, bem como, um indicador de como a ordenação deve ser realizada. As entidades especializadas *DatabaseNode* e *DatabaseAttribute* representam, respectivamente, um elemento e um atributo da XML.

A ordenação dos elementos e/ou atributos é definida utilizando a enumeração *OrderModifier*. Os valores que representam a ordenação são *ascending* e *descending*. Cada entidade *Database* pode conter ou não um ou mais elementos *constraints* que são instâncias da entidade *ExpressionVix*, que, por sua vez, podem ser instâncias das entidades *Equals*, *Different*, *LessThan*, *GreaterThan*, *LessThanOrEqual*, e *GreaterThanOrEqual*. Os elementos *constraints* são utilizados para adicionar restrições às consultas.

A entidade *ExpressionVix* é utilizada como operador nas comparações entre instâncias da entidade *Node*. As entidades *ValueNode* e *Database*, que são especializações da entidade *Node*, representam, respectivamente, um valor concreto definido pelo usuário e um elemento ou atributo XML. A entidade *ValueNode* representa seu valor através do atributo *val*. A Figura 3‑6 apresenta um exemplo de um modelo em conformidade com o metamodelo da ViXQueL.

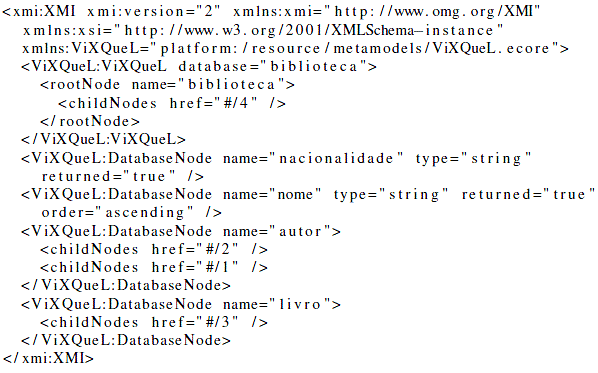


Figura ‑ – Exemplo de um modelo da ViXQueL

O metamodelo da linguagem XQuery tem como elemento principal a entidade *XQueryProgram* que representa uma consulta. A entidade *XQueryProgram* contém um elemento *command*. Este é uma instância da entidade FLWOR (3) (acrônimo para *For*, *Let*, *Where*, *Order**By*, *Return*). Para representar a funcionalidade completa de um comando FLWOR, a entidade de mesmo nome contém as entidades *Clause*, *Where*, *OrderBy* e *Return*.

A entidade *Clause* permite representar as cláusulas *For* e *Let* da construção FLWOR da XQuery. Para isto, foram definidas três especializações para a entidade *Clause*, que são: *For*, *Let* e *ClauseSequence*. A entidade *ClauseSequence* contém os elementos *clause1* e *clause2*, permitindo assim a definição de mais de uma cláusula (*For* ou *Let*). A entidade *Clause* contém ainda um elemento *variable* que é uma instância do elemento *Declaration*.

A entidade *Declaration* permite definir as variáveis e seus respectivos caminhos na consulta. Foram definidas, para isto, duas especializações para a entidade que são: *VariableDeclaration* e *DeclarationSequence*. A entidade *DeclarationSequence* foi definida com o intuito de permitir a declaração de mais do que uma variável. Isto se torna possível através dos seus dois elementos, *declaration1* e *declaration2*, que são instâncias de *Declaration*.

A entidade *VariableDeclaration* é a entidade que de fato permite a declaração de uma variável na consulta e contém dois atributos para representá-la: o nome e o tipo. Esta entidade contém ainda um elemento *IN* que é uma instância da entidade *Expression* e representa o caminho ou consulta FLWOR onde a variável será associada. Os tipos das variáveis são definidos utilizando a enumeração *DataType*. Os valores que representam os tipos das variáveis são: *string*, *integer* e *date*.

Após a entidade *Clause*, foi definida a entidade *Where*. Esta entidade contém um elemento *expression* que é uma instância da entidade *Expression* que é utilizado na representação das restrições da consulta. A entidade *FLWOR* contém também um elemento *where* que é uma instância da entidade *Where*, bem como um elemento *orderBy* que é uma instância da entidade *OrderBy*. Esta última entidade representa a ordenação da consulta e contém um atributo *order-Modifier* para definir a forma de ordenação. A entidade *OrderBy* contém ainda um elemento *exp* que é uma instância da entidade *Expression* com o intuito de definir qual será o elemento que será ordenado. A ordenação dos elementos e/ou atributos é definida utilizando a enumeração *OrderModifier*. Os valores que representam a ordenação são *ascending* e *descending*.

A entidade *FLWOR* contém ainda um último elemento *return* que é uma instância da entidade *Return* e representa o retorno da consulta. A *Return* contém um elemento *expression* que é uma instância da entidade *Expression* com o intuito de definir quais serão os elementos retornados pela consulta.

Por fim, as entidades *SequenceExp*, *BinaryExp*, *UnaryExp* e *FLWOR* são especializações da entidade *Expression*. A entidade *SequenceExp* contém os elementos *exp1* e *exp2* e tem o intuito de permitir a declaração de uma seqüência de expressões.

A entidade *BinaryExp* contém os elementos *left* e *right* e tem o intuito de permitir a declaração de operações binárias. Ela é utilizada como operador entre elementos e pode assumir uma instância de qualquer uma de suas especializações: *EqualsExp*, *DifferentExp*, *LessThanExp*, *LessThanOrEqualExp*, *GreaterThanExp*, *AndExp*, ou ainda, *GreaterThanOrEqualExp*.

A última especialização da entidade *Expression* é a entidade *UnaryExp*. As entidades *ValueExp* e *XPathExp* são especializações da entidade *UnaryExp* e representam, respectivamente, um valor literal especificado pelo usuário e um caminho de um elemento. A entidade *ValueExp* contém dois atributos para representar um valor literal, são estes: *val* e *type*. Para representar um caminho, a entidade *XPathExp* se constitui de um elemento *node* que contém um atributo *path* armazenando o caminho. A Figura 3‑7 apresenta um exemplo de um modelo válido para o metamodelo da XQuery.

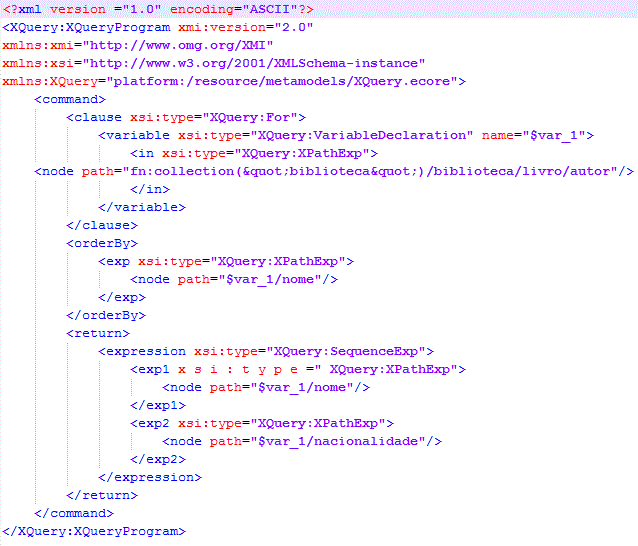


Figura ‑ – Exemplo de um modelo da XQuery.

## Transformações entre modelos

O mapeamento das entidades do metamodelo da ViXQueL para seus correspondentes na linguagem XQuery é feito usando uma linguagem de transformação de modelos. Durante o desenvolvimento da ViXQueL foram avaliadas duas dentre as mais conhecidas: Kermeta (37) e ATL (38).

**Kermeta** é uma linguagem orientada a modelos, imperativa, orientada a objetos e com verificação de tipos estática. Entre os conceitos tipicamente orientados a modelos estão associações, multiplicidades e gerenciamento de conteúdo dos objetos. Sua sintaxe foi inspirada pela linguagem *Eiffel* (39) e sua execução é realizada por um interpretador.

**ATL** é uma linguagem híbrida que incorpora tanto o paradigma imperativo quanto o paradigma declarativo. Entretanto, o estilo preferencial é o declarativo. Código imperativo deve ser usado para especificidades de mapeamento que são difíceis de descrever em uma forma declarativa. Embora possua construções imperativas, alguns problemas de transformação são relativamente difíceis de resolver como, por exemplo, o mapeamento de muitos elementos para muitos elementos. Seu código é compilado para um bytecode e sua execução é realizada por uma máquina virtual própria.

A linguagem escolhida foi a Kermeta. Isso não se deu apenas pelo fato dela ser mais indicada para realização de transformações de maior complexidade, bem como por ser uma linguagem que permite descrever a estrutura e comportamento dos seus modelos. Além dos fatos já mencionados, a Kermeta é compatível com a Ecore e com a linguagem de metamodelagem EMOF, um fragmento do MOF 2.0. Com todas essas vantagens, ainda é possível executar o código Kermeta dentro de um código Java(40).

O primeiro passo do processo da transformação da consulta visual é a chamada de um método Javascript que varre as informações previamente criadas na GUI e gera uma string contendo o modelo ViXQueL da consulta criada. Este modelo representa o PIM de uma consulta a um SGBD XML nativo e deve estar em conformidade com o metamodelo ViXQueL (**ViXQueL.ecore**).

Ao fim da geração do modelo, através de uma requisição AJAX, a própria função realiza uma chamada ao método estático *generate* da classe *CodeGenerator*, passando como parâmetro a string que representa o modelo ViXQueL. Este método é o responsável por realizar todo o trabalho de transformação da string representativa do modelo ViXQueL para o código XQuery.

O método *generate* realiza cinco passos para a geração textual do código XQuery. Em primeiro lugar, é criado um arquivo com o conteúdo do modelo ViXQueL (userViXQueL\_identArquivo.xmi). O *identArquivo* no nome do arquivo serve para determinar unicamente o arquivo utilizado.

Após a criação do primeiro arquivo, é invocado o método *execute* da classe *ViXQueL2XQueryTransformation*, passando como parâmetro a chave *identArquivo* gerada. Neste último método, é preparado o ambiente de execução para o Kermeta realizar a transformação definida em “ViXQueL2XQuery.kmt”, sendo esta a responsável pela transformação de PIM para PSM.

Após as transformações, um novo arquivo “userXQuery\_identArquivo.xmi” é gerado. Este modelo representa o PSM de uma consulta e deve corresponder ao metamodelo XQuery (XQuery.ecore). Este modelo descreve detalhes de como cada construção será implementada no modelo textual XQuery.

Em caso de sucesso na geração desse segundo arquivo, é invocado o método estático *execute* da classe *XQuery2CodeTransformation*. Este novo método se encarrega de executar as transformações definidas no “*XQuery2Code*.*kmt*”, neste momento, obtendo-se a transformação PSM para Código.

O código Kermeta existente na transformação “*XQuery2Code.kmt*” carrega o arquivo “*userXQuery\_identArquivo.xmi*”, varre o modelo realizando as transformações necessárias para a geração do código XQuery e, ao final do seu processamento, salva o código obtido como “*user-Code\_identArquivo.xquery*”.

Para finalizar este grande processo, o método *generate* armazena o código gerado em uma string, remove os arquivos temporários que foram criados para auxiliar nas transformações (*userViXQueL\_identArquivo.xmi*, *userXQuery\_identArquivo.xmi* e *userCode\_identArquivo.xquery*) e retorna para o Javascript a consulta XQuery gerada. Então, com o código XQuery criado, a base de dados XML já pode ser consultada. A Figura 3‑8 mostra o núcleo do processo de geração da consulta XQuery dividido dentre as camadas MOF.

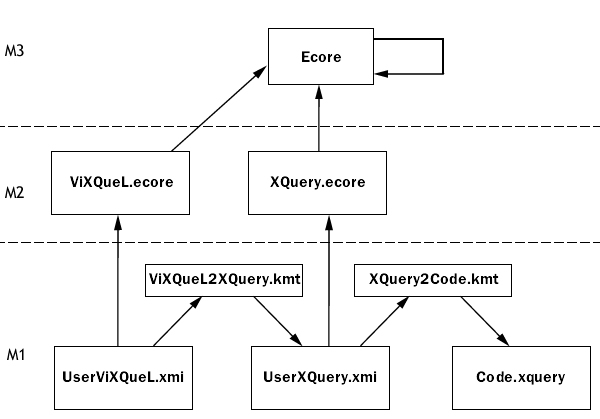


Figura ‑ – Processo de transformação da ViXQueL para código XQuery

Por fim, vale salientar que, devido a todas essas transformações realizadas no processo de geração da consulta serem executadas automaticamente pela aplicação residente no lado do servidor, estas são transparentes para o usuário, que não precisa ter qualquer conhecimento deste precesso.

## Expressividade da ViXQueL

Diferentemente de alguns trabalhos relacionados, a ferramenta **ViXQueL** não foi projetada para replicar toda a funcionalidade da linguagem XQuery, visto que a principal preocupação é que possa ser usada por não-especialistas, que dificilmente saberiam utilizar toda a expressividade da linguagem. O Quadro 3‑2 apresenta uma listagem de algumas funcionalidades da XQuery e uma indicação se estão presentes na ViXQueL ou não.

É importante ressaltar que algumas destas funcionalidades poderiam ser adicionadas à ferramenta sem grandes mudanças nos metamodelos e/ou transformações. Pode-se citar como exemplo destas funcionalidades: quantificação existencial, junção e agregação. No entanto, a adição de tais funcionalidades traria mais complexidade ao uso da ferramenta, necessitando assim de uma avaliação de impacto da usabilidade.

Quadro ‑ – Funcionalidades da XQuery suportadas pela ViXQueL.

|  |  |
| --- | --- |
| Funcionalidade XQuery | ViXQueL suporta? |
| Conjunção | Sim |
| Projeção em largura | Sim |
| Projeção em profundidade | Sim |
| Ordenação | Sim |
| Junção | Sim |
| Quantificação existencial | Não |
| Agregação | Não |
| Negação | Não |
| União | Não |
| Computações aritméticas | Não |
| Novo elemento | Não se aplica |
| Renomeação | Não se aplica |

Além das funcionalidades que podem ser adicionadas, existem as funcionalidades que não se aplicam à ferramenta aqui proposta. Tais funcionalidades não se aplicam devido ao fato de que a consulta é construída totalmente baseada em elementos existentes e pré-definidos, impedindo propositalmente que o usuário defina algo novo, assim garantindo que as consultas sempre serão válidas e utilizarão elementos existentes. São exemplos desta categoria de funcionalidades: Novo elemento e Renomeação.

A expressividade da VixQueL é limitada, se comparada com a expressividade total da XQuery. Entretanto, por se tratar de um protótipo, para validação da proposta do trabalho de criação da ViXQueL, foi necessário reduzir o escopo afetando, assim, a expressividade da ferramenta.

## Considerações Finais

Este capítulo abordou o protótipo inicial da ferramenta ViXQueL, proposta para abstrair o acesso às informações contidas em bases de dados XML nativas e foi dividido em três partes. Primeiramente, foi apresentada a arquitetura da ferramenta que foi implementada utilizando três camadas. Além disso, foi apresentado seu diagrama de classes para permitir um maior entendimento do projeto e implementação da ferramenta.

Posteriormente, foi abordado o sistema em si e suas funcionalidades. Foi mostrado, através das telas do sistema, como é possível controlar o servidor, iniciando-o e parando-o, além de controlar as bases de dados permitindo, iniciá-las, pará-las, criá-las e destruí-las. Foi apresentado também como é possível carregar arquivos XML nas bases de dados.

Em seguida, foi detalhado o processo de geração da consulta, desde o momento que o usuário constrói a consulta visual até a realização da consulta textual em um SGBD XML nativo. Para que este processo aconteça, foi necessário definir dois metamodelos e dois programas Kermeta para transformar modelos.

Por fim, foi discutido o poder de expressividade da ViXQueL apresentando uma lista com algumas funcionalidades da XQuery e indicando se estão presentes na ViXQueL ou não. Além disso, foi discutido o porquê de algumas funcionalidades não estarem presentes na ferramenta.

# Colaboração

“O futuro pode ser definido em uma palavra: colaboração.” (Jean Paul Jacob, IBM, 2008)

“Ato de colaborar; cooperação; ajuda. Trabalho feito pelos colaboradores. Reunião de duas ou mais pessoas que trabalham juntas para produzir ou utilizar uma aplicação.” São apenas algumas definições de colaboração.

Por sua vez, aplicativos colaborativos são aqueles os quais dão suporte ao trabalho em grupo. Além disto, uma de suas características mais importantes é a de criar a possibilidade de que pessoas com objetivos convergentes, mas em lugares remotos distintos, comuniquem-se e cooperem entre si. Alguns exemplos de *software* colaborativos são *webmail*, calendário, *instant* *messaging*, *wiki* e *bookmarking*. Mas, aplicativos colaborativos não são apenas aqueles os quais permitem uma colaboração em um nicho empresarial. Também sendo considerados como ***groupware*** (um neologismo criado para denominar os aplicativos voltados ou que possibilitam o trabalho em grupo) aplicações como, por exemplo, redes sociais.

Um ambiente de trabalho colaborativo, então, pode oferecer suporte aos seus integrantes tanto num âmbito individual quanto num âmbito cooperativo. Observa-se, com isto, o surgimento de um novo tipo de profissional: o *e-professional*, que são aqueles que trabalham juntos, colaborando entre si independentemente da sua posição geográfica.

Essa noção de *software* colaborativo está intimamente ligada à noção de sistemas de trabalho colaborativo. Este último remete a qualquer forma de organização humana de colaboração independente do tempo em que ocorreu e de sua forma: formal ou não, intencional ou não (41)**.**

## História

A colaboração pode ser encontrada desde quando o homem passou a criar meios de se comunicar para executar tarefas com o mesmo objetivo final. Podem ser citadas como grandes marcos da comunicação e, por conseguinte, da colaboração, ferramentas como a escrita cuneiforme (3400 a.C.), a prensa móvel (Gutenberg, 1450), a fotografia (Daguerre, 1839), o telefone (Bell 1876), filmes (Lumiere, 1895), rádio (Tesla, 1891 ou Marconi, 1895) e a vídeo-conferência (Bell PicturePhone, 1956).

Não menos importante, posteriormente às ferramentas citadas, foi criada a ARPANET (John Postel, David Crocker e Vint Cerf, 1969). Sua utilização culminou no aparecimento de novas formas de colaboração, tais como: o *e-mail* (Ray Tomlinson, 1971), a interface gráfica do usuário (~1984) – sendo considerada a usabilidade um fator de suma importância para o sucesso da colaboração, e a *World Wide Web* (Tim Berners-Lee @ CERN, 1990). Esta última, por sua vez, se tornou a ferramenta de comunicação e colaboração mais significativa do período do fim do século XX e início do século XXI.

Um dos primeiros registros do uso do conceito *groupware* é da década de 1980 e se deu numa declaração de Richman and Slovak(42) sobre uma nova forma de encarar o uso dos computadores: "*Como um nervo eletrônico que vincula as equipes em conjunto, o novo groupware visa colocar o computador no meio de comunicação entre gestores, técnicos e qualquer outra pessoa que interage em grupos, revolucionando a forma como eles funcionam.*"

E, já no inicio da década de 1990, uma das maiores empresas de tecnologia da época, a IBM, lança o **IBM Lotus Notes** (43). Esta foi uma das primeiras ferramentas criadas com base na colaboração e tinha por característica unir os serviços de email, agenda e de outras aplicações da própria IBM numa arquitetura cliente-servidor. Mesmo com a internet ainda na sua fase prematura, já era de notória importância serviços que posteriormente estariam disponíveis na grande rede.

Por este motivo, dois renomados estudiosos da área de tecnologia, David Kirkpatrick e Stephanie Losee, em 1992, previram uma grande mudança da definição de ambientes de trabalho como, por exemplo, um escritório. Atribuindo a essa drástica alteração o novo conceito de colaboração e sua contribuição no nível de produtividade em longo prazo. “*Você será capaz de trabalhar eficazmente como membro de um grupo onde você estiver com seu computador. E, como os computadores se tornam menores e mais poderosos, isto significará em qualquer lugar*.” Em adição a isto, afirmaram, também, que esse novo paradigma na época permitiria aos participantes de um *brainstorm*, por exemplo, em volta de uma mesa falassem de uma vez só. E o resultado dessa mudança seria um ganho imensurável de produtividade.

Os *groupware* com o passar do tempo se transformaram nos *software* empresariais de maior visibilidade e procura. Em uma pesquisa recente, 98% dos CEO ­– sigla para, do inglês, *Chief Executive Officer* – afirmaram que a prioridade das pesquisas e inovações dentro das médias e grandes empresas é constituída por ferramentas voltadas ao campo da inovação. Além disso, foi constatado que 90% dos funcionários trabalham fora da matriz da empresa e 60% exercem partes de suas funções em lugares diferentes dos seus gerentes (44). A crescente concorrência mundial impulsiona as grandes empresas a procurarem meios de colaboração e inovação para suas equipes. Estas, por sua vez, estão cada vez mais distribuídas geograficamente.

E, com a finalidade de se transformarem em empresas mais competitivas e inovadoras, há atualmente uma tendência de implementação de software para oferecer mais formas de colaboração. A mensagem instantânea (*chat*), por exemplo, permite a comunicação e, por conseguinte, uma colaboração em tempo real. Por isso, esta é comumente agregada às demais aplicações e processos existentes. As ferramentas orientadas ao desenvolvimento de comunidades proporcionam, também, aos seus usuários um acesso instantâneo e seguro aos conteúdos compartilhados.

O fato é que os sistemas fechados interferem na agilidade do negócio. Ao aproveitar ao máximo software da atualidade e integrar os novos processos à infra-estrutura existente, pode-se construir ferramentas mais aperfeiçoadas e próximas ao objetivo cooperativo.

## Tipos de Colaboração (tempo x espaço)

Como foi mostrada anteriormente, a colaboração está intrinsecamente ligada ao conceito de comunicação e à forma de como essa transferência de informação é realizada. Com a finalidade de classificar esses tipos de comunicação, a faz uma relação entre as variáveis **espaço** e **tempo**. São gerados, desta forma 4 (quatro) possíveis tipos de colaboração:

O primeiro tipo (a) mostra a colaboração entre duas pessoas de forma assíncrona mas que necessita a presença física de uma delas. Esta colaboração pode ser exemplificada, por exemplo, com as pinturas rupestres que passam, através dos séculos, as informações da época nas quais foram criadas. Obviamente, para poder ver estas informações criadas, uma pessoa necessita estar presente fisicamente no local onde as pinturas foram criadas.

Na segunda situação (b), a colaboração é realizada de forma síncrona e necessita, também, da presença física dos seus participantes. Este segundo tipo pode ser exemplificado com uma simples inspeção de código onde os participantes da inspeção precisam estar simultaneamente em frente a um mesmo computador. Ou ainda, uma reunião em um escritório, onde todos os integrantes desse encontro estão fisicamente em volta de uma mesa no mesmo horário.

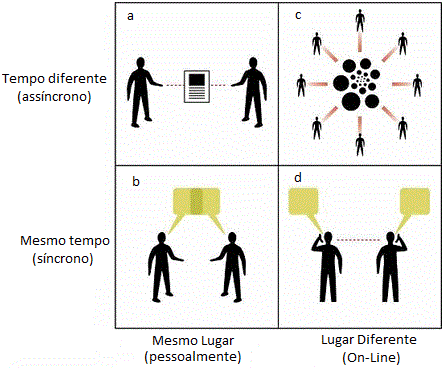


Figura ‑ – Tipos de colaboração: tempo x espaço – traduzida de (45).

O terceiro tipo (c), por sua vez, representa um tipo de colaboração onde a informação pode ser vista por várias pessoas a qualquer momento e em qualquer lugar. Para isso, seus integrantes precisam de algum meio onde possam acessar essas informações. Como exemplo desse modelo de comunicação ou colaboração, pode-se considerar um blog, uma “wiki” (46) ou um fórum. Nessas aplicações os membros participantes acessam, através da internet, informações que foram construídas por diversas pessoas, em vários lugares do mundo, durante anos.

E, o quarto e último tipo (d), expõe uma colaboração onde os seus personagens podem estar em qualquer lugar do mundo (inclusive ao lado), mas utilizam um meio de comunicação a distância. Esse quarto tipo (d) se difere do terceiro (c) por exigir a presença simultânea dos integrantes apesar de estarem em diferentes posições geográficas. Um exemplo típico desse paradigma é um chat textual on-line ou um VoIP (47).

A proposta desde trabalho, então, é adicionar características que atualizem a ViXQueL tornando-a mais colaborativa. Para isto, foi adicionado a ela funcionalidades tais como um chat para comunicação entre as pessoas que estão colaborando entre si na construção de uma mesma consulta (tipo d da Figura 4‑1). Outra funcionalidade acrescentada a ViXQueL foi a possibilidade de um usuário salvar uma consulta, construída ou em processo de construção, a fim de um posterior aprimoramento da mesma ou para sua recuperação em outras ocasiões. As funcionalidades de colaboração, bem como as conseqüentes mudanças na arquitetura da ViXQueL, serão mais detalhadas na próxima sub-sessão.

## Arquitetura da Colaboração

Conhecendo o significado da colaboração, um pouco de sua história, e dado a sua grande importância, este trabalho propõe a agregação de ferramentas que adicionem características colaborativas à ViXQueL.

Desta forma, foi adicionada à ferramenta a funcionalidade de salvar consultas em qualquer que seja seu estágio. Pode-se, assim, recuperar consultas já realizadas e salvas anteriormente. Inclusive, é permitido a um usuário carregar consultas criadas por outros usuários e com isso aumentar seu leque de conhecimento sobre a base de dados utilizada e/ou sobre a utilização da ferramenta. É possível observar, então, que essa primeira característica é do tipo (c) assíncrona/*On-Line*.

A segunda contribuição à ViXQueL foi dar a possibilidade de um grupo de usuários se reunirem remotamente numa mesma sala virtual (*Room*) com a finalidade de construir uma consulta ViXQueL a uma base de dados comum entre eles. Nesta sala virtual, apenas um dos componentes do grupo tem a permissão para alterar a consulta em andamento. Além disso, de modo a facilitar a comunicação em uma sala virtual, foi adicionado um espaço para mensagens instantâneas (*chat*) onde todos os usuários poderem conversar entre si. E, como conseqüência desta terceira contribuição, não somente os usuários poderão saber quais os outros usuários presentes na sala, mas poderão discutir qual a forma mais adequada para a criação da consulta em relação aos dados desejados.

Obviamente, para usufruir da segunda e terceira contribuição, os participantes da colaboração devem estar presentes em um mesmo momento, mas não precisam necessariamente estar num mesmo local físico. Dessa maneira, podem-se classificar essas últimas características adicionadas como do tipo (d). A arquitetura dessas novas funcionalidades está bem ilustrada na Figura 4‑2:

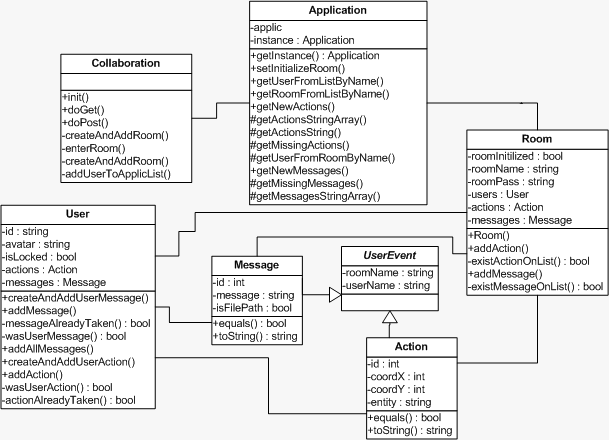


Figura ‑ – Modelo da colaboração.

Na Figura 4‑2, pode ser vista a arquitetura responsável pela colaboração no lado do servidor da ViXQueL. A classe “*Collaboration*” estende a classe abstrata “*HttpServlet*” do pacote “*javax.servlet.http*” e recebe requisições – *doGet*() – do lado cliente da aplicação. As requisições realizadas a esta classe podem ter a finalidade de: criar uma “*Room*”, criar um “*User*”, executar uma entrada/saída de um “*User*” de uma “*Room*”, entre outras.

A classe “*Collaboration*” possui uma instância estática de uma “*Application*” que, por sua vez, possui uma referência para a interface “*ServletContext*” do pacote “*javax.servlet*”. Assim, o contexto é guardado por toda a execução da ferramenta, desde sua configuração até a criação da consulta por parte do usuário. Neste mesmo “*ServletContext*” é armazenado os elementos “*Room*” criados.

Uma “*Room*” representa uma sala virtual onde “*User*” podem entrar e sair, bem como interagir com seus elementos. Cada uma dessas salas, por sua vez, possui uma lista de “*User*” referente aos usuários cadastrados na sala, uma lista de “*Action*” que corresponde ao conjunto de ações tomadas por todos os usuários dentro da sala e, por fim, uma lista de “*Message*” indicando toda a conversa ou todas as falas de todos os usuários dentro da sala virtual.

Um “*User*” representa um usuário da ViXQueL e, assim como uma “*Room*”, armazena uma lista de suas “*Action*” e “*Message*”. As listas servirão para uma verificação contínua de atualização entre as listas da “*Room*” e a lista de cada “*User*”. As listas dos usuários devem estar sempre em consistência em relação às listas da sala virtual. Caso isso não seja verdade, as modificações recentes na sala deverão ser replicadas nas listas dos seus usuários e, por sua vez, na suas *views*: interfaces gráficas no lado cliente da aplicação.

Cada “*Room*” só poderá ter um “*User*” **ativo** por vez. Apenas esse usuário ativo tem permissão para adicionar/remover entidades à consulta enquanto os demais usuários presentes na sala conseguem visualizar as ações realizadas até então. Sendo permitida somente a interação com outros usuários através da área de mensagem instantânea. Para haver mudança do usuário **ativo** é necessário que um dos integrantes **não-ativos** requisite essa transferência de status. Cabe, então, ao usuário ativo, aceitar a requisição ou não. Se a requisição for aceita, o usuário até então **ativo** passa para o estado de **não-ativo**, enquanto o requisitante tem seu estado alterado para **ativo**. Caso contrário, os estados dos participantes na sala permanecem inalterados.

Desta forma, uma sala virtual para criação de consultas a um banco de dados XML nativo é criada. E, sob essa arquitetura, representa-se também os usuários e suas possíveis interações com a parte colaborativa da ViXQueL. Assim, permite-se que usuários com propósito comum compartilhem informações sobre a criação de consultas. Como exemplo dessa interação, pode-se citar um professor que deseje ensinar um grupo de alunos a criar uma consulta visual e em seqüência visualizar a consulta XQuery gerada, ou ainda, uma usuário que deseja reutilizar consultas anteriormente criadas afim de analisar as novas atualizações nos dados.

## Funções Adicionadas

Como já foi dito anteriormente, as funções adicionadas à ferramenta ViXQueL relacionadas à colaboração foram: a adição de um chat dentro da ferramenta, o salvamento de consultas para recuperação posterior da consulta e a possibilidade de visualização da construção da consulta visual por usuários distintos. Nas subseções seguintes, serão mostradas as novas telas da ferramenta, bem como as novas funções que agregaram valor à ferramenta ViXQueL.

## Novas Telas da ViXQueL

Antes de explicar as principais funcionalidades adicionadas à ferramenta ViXQueL, nessa subseção, serão mostradas as telas as quais o usuário terá que passar para chegar na nova tela de criação de consulta.

A primeira tela () mostra os campos de nome e o avatar escolhido pelo usuário. Esse usuário não conseguirá passar para próxima tela caso haja um segundo usuário com o mesmo nome. E seu avatar é inicialmente escolhido randomicamente, mas pode ser alterado caso o mesmo deseje.

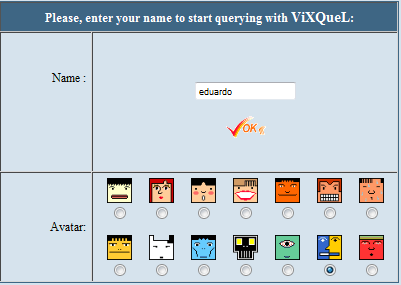


Figura ‑ – Tela inicial da ViXQueL

A segunda tela (Figura 4‑4) exibe as salas (*rooms*) existentes, bem como apresenta os campos de criação de uma nova sala. Nessa fase da interação com a ferramenta, o usuário pode visualizar as salas existentes: *consulta02* e *consulta03*; bem como os demais usuários presentes em casa sala. Para entrar em uma sala, o usuário deve clicar no botão “*enter*”.

Caso deseje, o usuário também pode criar uma sala nova. Ele deve entrar com o nome da nova sala e apertar no botão “*create*”. Neste segundo caso, o usuário ainda deverá escolher qual base consultar e inicializá-la, exatamente como era realizado na versão anterior da ViXQueL. Após esses passos serem realizados, o usuário entrará na tela de criação de consulta com as novas funções de colaboração.

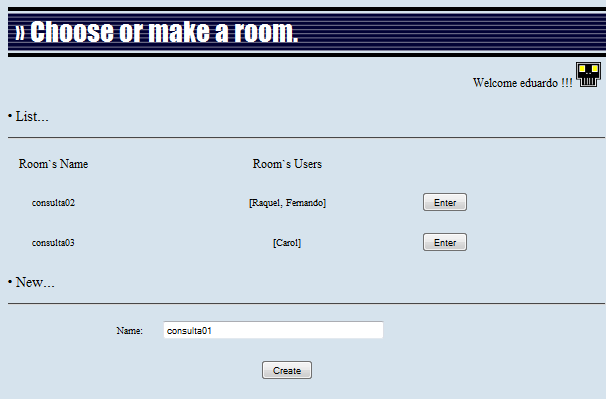


Figura ‑ – Escolha/Criação de sala.

Esta nova tela de consulta após a adição das funcionalidades é mostrada na Figura 4‑5:

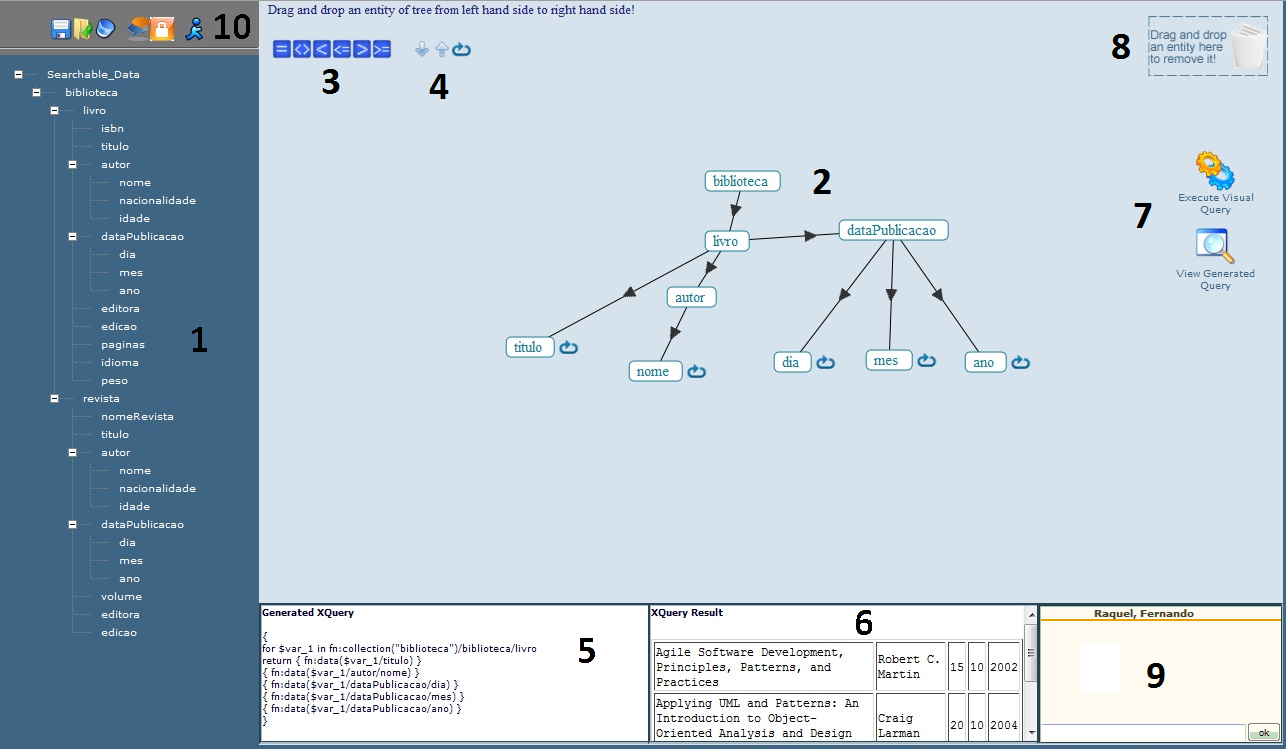


Figura ‑ – Nova tela de consulta da ViXQueL

Nas subseções seguintes , as três principais funções adicionadas à ferramenta ViXQueL, serão explicadas mais detalhadamente.

## Chat

Um chat dentro da ferramenta permitirá que o usuário mantenha mais o foco na aplicação durante a criação das consultas visuais. Com esta funcionalidade, as pessoas envolvidas no processo de criação poderão ter uma maior noção do ambiente e suas variáveis, tais como: visualizar a consulta que está sendo criada no momento; visualizar a estrutura da base consultada; bem como debater com os outros participantes da sala a melhor forma de utilizar a ferramenta e/ou criar as consultas. Essas novas funções são permitidas, sobretudo, sem que o usuário necessite realizar operações de mudanças de tela.

Para inserção do chat na ferramenta, algumas modificações estruturais na interface visual da tela de criação de consulta () foram necessárias. Foram reduzidas as dimensões dos campos 5 (cinco) e 6 (seis): “Consulta XQuery gerada” e “Resultado da consulta”, respectivamente. Desta forma, o chat foi posicionado da extrema direita desses campos. Os itens de limpar a tela (*“Start New Query”)* e sair (“*Exit”)* do campo 7 (sete) foram movidos para o novo menu Figura 4‑6 adicionado acima do campo 1 (um), visando-se, assim, liberar o espaço vertical do chat.

## Salvar/Recuperar consultas

A nova funcionalidade de salvar e recuperar consultas criadas, apesar de ser auto-explicativa, é de suma importância para a realização de atividades colaborativas, tais como: evitar a necessidade de criação de uma mesma consulta mais de uma vez por usuários; realizar a correção de consultas criadas por alunos em atividades de ensino à distância; entre outros.

Os botões referentes às atividades de salvas e recuperar as consultas foram adicionados no novo menu (Figura 4‑6) mencionado anteriormente. Para a operação de salvamente da consulta criada, é necessário que pelo menos uma entidade da consulta esteja presente na área de criação de consultas campo 2 (dois) da Figura 3‑4.



Figura ‑ – Menu de ações: Salvar, Abrir, Limpar tela, (des)habilitar chat,

(un)lock e sair, respectivamente.

## Visualização da criação da consulta em tempo real

A visualização da criação da consulta em tempo real por usuários geograficamente separados é a última, e não menos importante, funcionalidade adicionada à ferramenta ViXQueL neste trabalho. Esta funcionalidade permite otimizar a criação de consultas visuais, uma vez que vários usuários presentes em uma mesma sala podem conversar entre si.

Esta funcionalidade, como já dito anteriormente, tem a característica de permitir apenas usuário ativo por vez. Ou seja, apenas um usuário na sala pode alterar a consulta em criação. Caso um usuário não-ativo deseje se tornar ativo, este deve fazer a requisição de *“lock”* e aguardar a autorização do usuário ativo. Esta requisição pode ser realizada através dos botões “lock” e “unlock” (Figura 4‑7) que, por sua vez se revezam entre si indicando o estado atual do usuário (ativo ou não-ativo).



Figura ‑ – Botões de *Lock/Unlock, respectivamente.*

Caso o usuário ativo aceite a requisição, mostrada na , o usuário não-ativo que requisitou o “*lock*” passa a ser o novo usuário ativo da sala.

Com essas 3 (três) novas funcionalidades principais, buscou-se aumentar o nicho contemplado pela ferramenta ViXQueL. A colaboração implantada na ViXQueL, por sua vez, permite que, não só usuários não-especialistas em XQuery (3) realize consultas a dados em SGBD XML nativos. Mas, também, que grupos de usuários com interesses comuns possam realizar consultas visuais e/ou aprender uma linguagem complexa como a XQuery.

Na próxima subseção, serão realizadas as considerações finais sobre os temas abordados no Capítulo .

## Considerações Finais

Neste capítulo, foram abordados inicialmente alguns conceitos sobre colaboração, bem como um pouco de sua importância nas diversas atividades do homem durante sua história.

Em seguida, foi mostrada uma classificação dos tipos de colaborações existentes relacionadas ao tempo e espaço dos seus integrantes. Observou-se que estes, por sua vez, podem estar presentes ou não em um mesmo local. Ou, ainda, podem colaborar ao mesmo tempo ou não. Também foi discutido qual as características de cada uma dessas classificações.

Posteriormente, foi apresentada a arquitetura adicionada à ferramenta ViXQueL visando a colaboração. Mostrou-se suas novas telas e as novas funções integradas a ferramenta. E, por conseguinte, foram exibidas as contribuições adicionadas à ferramenta ViXQueL a partir dessa nova arquitetura.

No próximo capítulo, será realizada uma validação heurística qualitativa da interface visual da ferramenta ViXQueL. Para isso, serão apresentados alguns conceitos sobre essa validação e sua importância.

# Validação

Nesta seção, serão mostrados o desenvolvimento da comunicação visual e a importância do design. Em seguida, será ratificado o valor da validação de interfaces gráficas e suas conseqüências. Posteriormente, será apresentada a metodologia utilizada para a validação da ViXQueL e os resultados obtidos através dessa validação.

## Interface gráfica

Os primeiros registros indicam que a comunicação humana foi inicialmente dada através de elementos visuais. Mesmo antes de o homem desenvolver a habilidade de se expressar mediante a linguagem falada, ele utilizou seu corpo para transmitir seus sentimentos, pensamentos e vontades.

Apesar disso, com o passar do tempo a linguagem falada passou a ser o meio de comunicação de informação mais direto, mas a linguagem visual continuou tendo uma grande importância nas relações. Em seguida, os materiais e suportes para “modelar” mensagens visuais vieram a corroborar com essa importância. Isso pode ser exemplificado – Figura 5‑1 – nas já mencionadas pinturas rupestres, nos hieróglifos, nos papiros e nos incunábulos ­– antigos livros manuscritos –, bem como nos mais atuais livros, cartazes publicitários, interfaces de usuário em modo texto e, finalmente, interfaces gráfica de usuário.

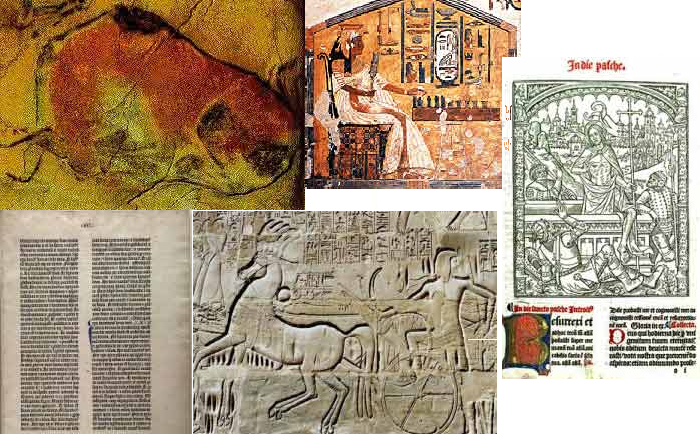


Figura ‑ – Pintura rupestre, papiro, incunábulo, livro e hieróglifo.

E o processo técnico e criativo relacionado à configuração, concepção, elaboração e especificação de artefatos, inclusive os visuais, é o *design*. Esse processo normalmente é orientado por uma intenção ou objetivo, ou para a solução de um problema (48). Estendendo esse conceito para o “*web design*”, pode-se afirmar que um site mal projetado pode levar a uma perda de rendimento ou produtividade.

A interface de um site, por sua vez, é uma mistura complexa de textos, *links*, elementos gráficos, tipos de formatações e outros aspectos que afetam a qualidade geral do site. Por conseguinte, *web design* site implica em um amplo conjunto de atividades de modo a satisfazer a estes diversos aspectos (49).

Assim, a questão de como melhorar o design de sites é de considerável importância. Essa preocupação aumenta quando se leva em conta os sites criados por não-profissionais da área. Ainda que a maioria das páginas *web* seja criada por empresas com profissionais de design, muitos sites de baixa visibilidade são criados por pessoas sem experiência ou treinamento. Como conseqüência disto, ainda se verifica uma quantidade substancial de sites com usabilidades abaixo do padrão (50).

Apesar de existir livros com guias sobre o *web design*, existe uma disparidade entre uma heurística de como criar uma interface consistente e a sua implementação. Além disso, esses manuais costumam levantar conceitos contraditórios como, por exemplo, ter um mesmo modelo para sites, independentemente dos seus propósitos ou público alvo.

*Interfaces web* desafiam as definições tradicionais de usabilidade. Um modelo de três fases de avaliação de um *website* é proposto, com base na atração inicial, navegação / exploração e operação (51). Usabilidade é redefinido como *trade-off* entre o aumento da motivação do usuário para incentivar a pesquisa e compra em *e-commerce*, por exemplo, e os custos dos erros de usabilidade. Para avaliar essa atratividade de interfaces de usuário *web*, heurísticas são propostas com base na concepção estética, a excitação geral criada por conteúdo, identidade corporativa e de marca, e a utilidade percebida adaptado às necessidades dos usuários.

As definições tradicionais de usabilidade têm enfatizado a utilidade e sua facilidade operacional (52). No entanto, as *interfaces web* estão fazendo muitos reverem essas definições tradicionais. Novos métodos de avaliação de usabilidade são concentrados em abordagens heurísticas para avaliar a qualidade do design (como por exemplo, Nielsen (53; 54; 55)) ou para detectar erros do usuário e inferir as causas dos defeitos (56).  A preocupação dos *web-designers* atualmente envolve mais perspectivas: o apelo estético e a atração de usuários. Se o usuário não for atraído a permanecer em um site, não importa quão bem projetada a usabilidade operacional pode ser. Como reconhecimento destas tendências, as variações de avaliação heurística surgiram para avaliar concepção estética ou de afinidade, ou seja, levar objetos à vida através do bom design visual (51).

Entretanto, a massiva maioria das validações de usabilidade que considera a satisfação do usuário ou a atratividade de uma interface simples utiliza questionários para captar votos sobre uma lista de variáveis ­– essas variáveis serão mostradas e discutidas em breve. Além disso, os projetistas têm pouco de orientação para a criação de interfaces de usuário atraentes. Alguns conselhos podem ser encontrados na comunidade de design visual / gráfica (57; 58). No entanto, esses conselhos são, em sua maioria, em forma de exemplos e cenários, deixando-os para serem interpretados e generalizados. Evidentemente, por essa característica, o modo como essas boas práticas são expostas cria brechas para interpretações variadas e que nem sempre condizem com a idéia original.

Por conseguinte, para avaliação de *websites* adota-se duas abordagens: primeiramente, a observação de erros ao usuário navegar nos sites, e a avaliação heurística em que a qualidade da *interface* é avaliada em função de uma lista de critérios escolhidos heuristicamente. Neste último caso, a eficácia das heurísticas foi avaliada para interfaces de usuário em geral por Nielsen & Molich (55) os quais relataram que 60% dos problemas observados podem ser identificados por especialistas usando a heurística. No entanto, essa validação de heurísticas quando usada como critério de qualidade em vez de guia de diagnóstico é menos precisa.

## Modelo de validação de interfaces web por usuários

A avaliação de usabilidade é tradicionalmente concentrada na facilidade operacional da utilização de uma interface (52). Para *interfaces web*, essa usabilidade tem sido focada na facilidade de navegação, clareza dos links, e a facilidade de busca de informação (por exemplo, Spool et al, (59)). Em adição a isto, métodos de avaliação heurística foram estendidos para *web*, no entanto, ainda sem chamar atenção para características de design que melhoram a previsibilidade e funcionamento intuitivo da interface. Enquanto estas são questões importantes, a eficácia global deve se preocupar também com a atratividade. No contato inicial com um *website*, a atração vai desempenhar um papel-chave e determinar o tempo de permanência (por quanto tempo um usuário olha para a *homepage*) e depois aumentar a motivação para a exploração. Somente após a decisão do usuário de permanecer em um determinado *site* ser tomada*,* as suas operações de usabilidade e navegação tornam-se questões dominantes.

Atratividade pode ser considerada a combinação do que motiva o usuário a navegar em um site. Assim, parte da problemática de projeto envolve a análise de requisitos e modelagem do usuário, resultando na escolha opcional de conteúdos para um conjunto de usuários. No entanto, existem também questões mais gerais, como o apelo estético de um projeto e as imagens *eyecatching* ­– mais apelativas no sentido de ter forte influência na atenção do usuário.

Além disso, existem outros efeitos no design que captam a atenção do usuário, como o uso de animações e som (60). Controle de atenção tem um complexo mecanismo cognitivo – ver (60) para mais detalhes sobre o assunto. Atratividade é influenciada não só pelo que já foi citado anteriormente, mas também pelas qualidades estéticas de um projeto, a motivação do usuário, suas necessidades e o tesão provocado pela interface.

Evidentemente, a conexão com o projeto pode ser mediada por muitas variáveis. Portanto, as fases do projeto precisam iniciar com a orientação dos requisitos e seleção de conteúdos, seguidos de seleção da mídia apropriada para representação desses conteúdos e, finalmente, o projeto final detalhado. Estas etapas da concepção estão associadas com o segundo conjunto de variáveis, começando com a escolha do material para informar ou estimular. Este d*esign* de formulário envolve a criação com fins estéticos, bem como selecionar os meios apropriados para a mensagem e, finalmente, design de efeitos voltados para a atenção do usuário (61).

O modelo das questões de design em um processo geral de interação com sites é ilustrado na Figura 5‑2. Supondo-se que o site de interesse do usuário foi encontrado, o objetivo da primeira fase é fazer o usuário permanecer no site durante um período suficiente para explorá-lo. A atração do usuário é indispensável nesta etapa. Atratividade pode ser medida pelo tempo de permanência registrado em sites, bem como avaliações do usuário em questionários, no entanto, é uma manifestação externa de três variáveis complexas (51):

* **Nível de Alerta** - é quanto um site parece estimulante ou não para o usuário. Essa característica está diretamente ligada ao interesse do usuário. Assim, seu nível quando moderado pode ser benéfico para maior disposição a exploração e/ou busca de resoluções para os problemas encontrados. Estas últimas características são de grande importância para a próxima fase;
* **Motivação** - é refletida na vontade de agir ou acreditar em algo, e tem uma teoria muito complexa na psicologia (62). A motivação é uma área pouco investigada no campo da IHC – Interação Humano-Computador – no que diz respeito às atitudes e predisposições para agir, dependendo de variáveis como a necessidade de bens, de poder, auto-estima e menos poderosos fatores de curiosidade, o aprendizado e o altruísmo; e
* **Observação de utilidade** - é influenciada diretamente pela necessidade de serviços ou bens, e indiretamente pelo poder, auto-estima e altruísmo. A utilidade requer um modelo detalhado que combine as necessidades do usuário com o produto ou serviço.

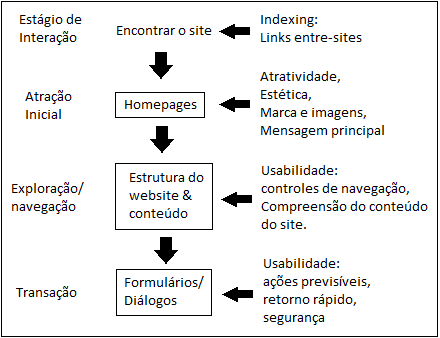


Figura ‑ – Padrão de atração e de usabilidade durante os estágios de interação com websites – traduzida de (51).

A primeira fase tem por objetivo ganhar a atenção do usuário. Para isso, são utilizadas técnicas baseadas nos diversos tipos de mídia. São exemplos disto: mídias dinâmicas como os vídeos ou sons, *banners* animados, textos destacados e os variados tipos de formatação das páginas *web*. Cada tipo de usuário tem uma resposta diferente às diversas mídias. E, para o sucesso delas, uma modelagem é necessária para especificar qual o tipo de usuário é alvo do *site*.

Uma vez que o usuário tenha sido atraído para o site e foi convencido a ficar, dá-se início à próxima etapa. Nesta fase de exploração e de navegação, a qualidade da usabilidade é imprescindível. Problemas com usabilidade podem determinar a continuidade da motivação do usuário para desbravar um site e, conseqüentemente, a sua interação com o mesmo. Ambigüidades, pistas enganosas sobre o site e a promessa de funcionalidades que na realidade não são oferecidas são exemplos de outros fatores que podem acarretar numa possível desistência de interação com o website por parte do usuário.

Na fase final, a busca dos benefícios ganha a prioridade. Na ferramenta ViXQueL, o usuário poderá procurar compreender melhor a estrutura da consulta visual e, mais adiante, a estrutura da linguagem XQuery. O uso intensivo de informações com um *layout* bem estruturado será de fundamental importância nesta fase. Outra técnica muito usada é o uso de linguagens e imagens de pessoas para gerar aspecto humano. Desta forma, pode-se mostrar interesse e atenção. É bem comum, também, se utilizar de figuras públicas ou de mulheres de boas feições para persuasão(63).

O projeto de um site de sucesso, portanto, deve buscar diretrizes de design diferentes para atender às variadas necessidades em casa fase. Para analisar as fases de interação, pode-se usar as seguintes técnicas e medidas:

* **Atração Inicial** - tempo de permanência, medido a partir de *logs* de interação do usuário e um questionário para investigar quais características os usuários perceberam que mais os atraíram ou repeliram. Testes de memória para estabelecer os tópicos do site que foram lembrados também podem ser utilizados;
* **Exploração e navegação** - medidas de usabilidade, como erros (%) e os tempos de conclusão da tarefa. Julgamento de especialistas sobre a conformidade com o *design* das orientações e heurísticas. Inspeções cognitivas de interação também podem ser usadas para identificar falhas de projeto; e
* **Operação** – são medidas de usabilidade como acima. Utiliza-se, além das medidas citadas, de entrevistas de esclarecimento para descobrir a avaliação dos usuários sobre o *website*.

Na sessão seguinte, será mostrado o processo de como a ViXQueL foi avaliada pelos seus usuários de acordo com as etapas vista anteriormente.

## Itens da Validação de ViXQueL

A atratividade pode ser dividida em qualidades genéricas de uma interface de usuário, tais como concepção estética e o uso de meios para direcionar a atenção. Bem como, questões relacionadas com o conteúdo de ligar o estilo visual, a marca e as mensagens para o conhecimento dos usuários sobre a organização. Para serem usadas tanto como modelo de design como critérios de avaliação, heurísticas são propostas. Tais heurísticas, por sua vez, podem estender práticas existentes no campo do design de *websites*, como, por exemplo, as heurísticas de Nielsen (54) ou os princípios do design da IBM (64). Por seguir uma abordagem experimental ao invés de uma filosofia de projeto de engenharia, esses princípios para a concepção estética se tornam controversos. Entretanto, alguns pesquisadores, citados anteriormente, formalizaram boas práticas para o design e os itens seguintes são baseados nas recomendações de suas heurísticas (57; 58):

* **Uso da cor** - o uso de cores deve ser equilibrado. Baixa saturação deve ser usada como cor de fundo. É aconselhável também não usar mais de 3 (três) cores de saturação intensa;
* **Simetria e estilo** - o *layout* deve ser simétrico. Por exemplo, a organização bilateral ou radial pode mostrar um jogo simétrico dos objetos. Formas curvadas também podem ser atrativas quando contrastada com formas retangulares;
* **Estrutura e consistência do *layout*** - uso de tabelas ou grades para estruturar componentes e retratar uma ordem consistente. Elas devem ser compostas de retângulos que não excedam a proporção 5/3 respectivamente para altura e largura;
* **Profundidade dos campos** - o uso de camadas numa tela estimula interesse e pode ser mais atrativa por promover um efeito de tranqüilidade. O uso de imagens com cores de baixa saturação provocam uma profundidade em relação aos objetos em primeiro plano;
* **Escolha de mídia para atrair atenção** - vídeo e sons têm um efeito de alertar e aumentar a atenção. A música pode atrair definindo um temperamento apropriado para o site;
* **Uso de personagens na mídia para persuasão** - este princípio é aplicado principalmente a sites de comércio eletrônico quando a utilização de imagens e vozes humanas pode ajudar a estimular o usuário a comprar mercadorias sendo educados ou elogiando suas escolhas;
* **Design de imagens intrigantes ou desafiadoras** - são estimulantes para a imaginação e criatividade dos usuários. Esse tipo de imagens exóticas com freqüência desobedece às leis de formas e perspectivas;
* **Estilo visual consistente** - esta heurística está na fronteira entre os dois conjuntos. O estilo visual é genérico no sentido de que um site deve ser consistente em termos de *layout* e imagem, mas o estilo também deve refletir os valores da empresa. Assim, um site orientado para o mercado jovem deve usar material mais estimulante, enquanto que um site voltado para usuários mais velhos podem usar um menos;
* **Visibilidade de identidade e marca** - a eficácia desta heurística depende da força da marca e da identidade corporativa. O princípio de construção apenas recomenda que a identidade esteja visível de forma coerente;
* **Correspondência ao humor do usuário e motivação -**esta heurística incide sobre a correspondência entre o tipo do usuário e o conteúdo do site. Algumas variações que podem ser esperadas são entre idade e sexo. Mas, em última análise, este é um tema complexo e é tratado em muitos livros como (65);
* **Despertar o interesse do usuário por motivação secundária -** atratividade pode ser aumentada pela adição de uma funcionalidade que não está orientada para a finalidade principal do site, mas pode atrair para uma motivação secundária. Alguns exemplos são a colocação de jogos e simulações em sites de comércio eletrônico para diversão dos usuários; e
* **Seleção/busca de conteúdo para atender às necessidades dos usuários** - isso deve resultar de uma profunda análise de requisitos. Deve-se tomar cuidado ao exibir conteúdos inadequados, pois isto pode confundir a análise de requisitos. Conteúdos relacionados às necessidades dos usuários devem ser claramente indicados, em linguagem inequívoca, com sugestões claras sobre como encontrá-lo.

Estas heurísticas precisam ser combinadas com os princípios de usabilidade existentes para dar uma noção da usabilidade e atratividade na avaliação. As heurísticas genéricas se aplicam mais intensamente para a fase de atração inicial de interação do *website*. Por sua vez, ambos os conjuntos se aplicam a atração inicial, exploração e operação de *sites*.  Se o site é bem avaliado sob as heurísticas acima, a motivação do usuário será mantida e alguns erros de usabilidade podem ser tolerados.

Na próxima seção, a heurística e o modelo de interação website são usados em uma avaliação do estudo de caso da ViXQueL.

## Validação da Ferramenta ViXQueL

Os participantes da avaliação classificaram a ViXQueL em uma escala de 1-10 para cada heurística e foram solicitados a relatar as razões da sua decisão e da facilidade com que cada heurística pôde ser interpretada. A soma das notas de cada integrante foi utilizada para verificação da pontuação de cada heurística.

Para isso, foi utilizada a ferramenta *online* *GoogleDocs* (66) na criação do formulário onde cada usuário deu sua nota e fez seus comentários sobre a ViXQueL acerca de cada um dos itens avaliados caso julgasse necessário. Esta mesma ferramenta gera um relatório sobre as resposta dadas com gráficos e textos.

A avaliação de cada usuário foi definida da forma em que se segue: Primeiramente, uma breve explicação em linguagem não-formal sobre o que é e qual o objetivo da ViXQueL.Em seguida, o objetivo do questionário, seus tópicos e como ele será efetuado. Esta primeira etapa pode ser vista com mais detalhes na Figura 5‑3.

Posteriormente, é mostrada a ferramenta e suas telas. Após isto, na tela de geração de consulta, é apresentada o Quadro 3‑1 – Funções das áreas da Ferramenta ViXQueL – vista no Capítulo 3.1, bem como é realizada uma consulta simples para o usuário ter uma noção de por onde começar a interação. Finalmente, deixa-se livre o usuário para experimentar a navegação na interface de geração de consulta.

No início da navegação, o usuário recebe 3 (três) descrições de possíveis consultas, que devem ser criadas durante sua avaliação. Bem como a estrutura dos dados a serem consultados. Essas consultas podem ser vistas também na Figura 5‑3. A complexidade dessas consultas cresce de forma gradativa, visando observar o desenvolvimento e capacidade de aprendizado de cada usuário. A complexidade citada se deve a quantidade de entidades envolvidas na criação da consulta visual. Dessa forma, pode-se constatar o nível de complexidade e intuitividade no uso da ViXQueL.

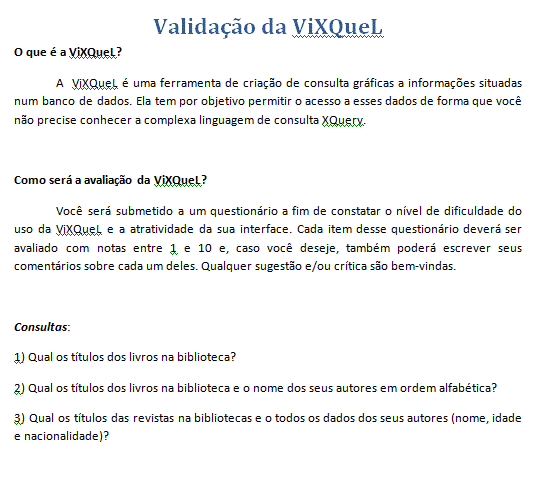


Figura ‑ – Instruções iniciais da avaliação.

Após a criação das consultas, o usuário em questão então recebe o questionário (Anexo I). O usuário continua com acesso a ferramenta para que ele investigue e reflita sobre os itens da avaliação. Em caso de dúvida, o avaliador tem total liberdade para discutir os itens. Ao término da avaliação, o usuário submete a resposta para a criação do relatório. Este relatório é mostrado na próxima seção.

## Resultados Obtidos

Os resultados que se seguem foram obtidos a partir da realização do questionário de validação com usuários não-especialistas em linguagens de consulta a sistemas de gerenciamento de banco de dados XML nativos. Ao todo, 7 (sete) pessoas com formações diversificadas foram submetidas ao processo de validação: 2 (duas) advogadas, 1 (uma) contadora cursando uma pós-graduação em moda, 1 (um) administrador com MBA em logística, 2 (Dois) estudantes de graduação em ciência da computação e 1 (uma) estudante do ensino médio.

Cada avaliação durou em média 20 a 25 minutos. E das notas dessa avaliação dos usuários, reunidas no *GoogleDocs*, foi tirada a média das notas de cada quesito. Essas médias estão exibidas na Tabela 5‑1. Com elas pode-se verificar a classificação de quais são os pontos fortes e fracos na usabilidade e atratividade de ViXQueL. E, como mencionado anteriormente, cada usuário teve a liberdade de comentar cada quesito analisado. Alguns exemplos dos comentários dados em cada quesito podem ser verificados a seguir:

Tabela ‑ – Média das notas da validação

|  |  |  |  |
| --- | --- | --- | --- |
| Uso da cor | Simetria e estilo | Estrutura e consistência do layout | Profundidade dos campos |
| 7,8 | 9,2 | 8,2 | 8,2 |
| Escolha de mídia para atrair atenção | **Uso de personagens na mídia para persuasão** | **Design de imagens intrigantes ou desafiadoras** | **Estilo visual consistente** |
| 4,6 | 1,5 | 4,8 | 6,4 |
| Visibilidade de identidade e marca | **Correspondência ao humor do usuário e motivação** | **Despertar o interesse do usuário por motivação secundária** | **Seleção/busca de conteúdo para atender as necessidades dos usuários** |
| 5 | 7 | 7,2 | 7,4 |
| Capacidade de ser intuitiva | **Dicas ao usuário (Hints)** | **A ViXQueL** |  |
| 9 | 6,2 | 7,8 |  |

* **Uso da cor** – “Deve usar um elemento contrastante, nos detalhes, não no todo para não tornar-se cansativo, mas não priorizar o cinza e tons de azul isoladamente. Usaria alguns itens de atenção em amarelo ou em vermelho. A uniformidade de cores acaba deixando o usuário perdido e cansado. As cores como amarelo e vermelho como são cores de atenção não devem ser usadas na maioria da página pois pode cansar também se usado em excesso.”. Ou ainda: “Podia escolher a cor como no orkut(67).”;
* **Simetria e estilo** – “Acho a simetria perfeita, pois no ocidente utilizamos a leitura da esquerda para a direita, mas se o público alvo for países orientais ou muçulmanos deve existir outra versão do aplicativo (apenas para comodidade do usuário).”;
* **Estrutura e consistência do layout –** “O resultado da consulta poderia ser mostrado num lugar maior.”;
* **Profundidade dos campos –** “O uso de camadas é realmente mais confortável”. Ou então, “Os objetos somem se forem botados muito pra cima. Foi preciso começar a fazer de novo.”;
* **Escolha de mídia para atrair atenção –** “Acho interessante a mídia como opcional, mas pode se tornar cansativo.”, “Sons atraem a atenção do público, porém música em sites corre o risco de se tornar cansativo”, “deveria ter, ia ser mais vistoso” e ainda “não há necessidade.”;
* **Design de imagens intrigantes ou desafiadoras –** “As imagens devem despertar algum atrativo, e deste modo a aplicação das cores deve ser colocada de acordo com a finalidade desta imagem para que não desperte mais atenção do que o objetivo para que foi criada” ou ainda “não vislumbrei esse tipo de imagem, também não acho adequado.”;
* **Estilo visual consistente –** “Considero que o estilo visual está adequado ao propósito da ferramenta.”;
* **Visibilidade de identidade e marca –** “O nome da ferramenta não aparece muito. Mas é bonito.” Ou ainda “Não fixou a marca (nome da ferramenta) na minha cabeça.”;
* **Correspondência ao humor do usuário e motivação –** “Desinteresse clássico referente ao assunto, embora tenha reconhecido a facilidade de uso do objeto em questão (principalmente para quem utiliza muito pouco coisas deste tipo. Provavelmente quando usasse, usaria este pela facilidade)”. Ou ainda, “Quando o publico alvo se identifica com a ferramenta aumenta o volume de acessos”;
* **Despertar o interesse do usuário por motivação secundária –** “Acho importante para ter o usuário por mais tempo no site fazendo multi-tarefas, como, por exemplo, atalho para acesso de email.”;
* **Seleção/busca de conteúdo para atender as necessidades dos usuários –** “Neste tipo de ferramenta não vejo necessidade de um dispositivo de busca.”;
* **Uso de personagens na mídia para persuasão –** “talvez seja aconselhável para auxiliar pessoas que não costumam utilizar este tipo de ferramenta.”, “Não tem personagens. Podia ter a Lady Gaga.”, “Algumas pessoas são formadoras de opinião, então o uso de personalidades para divulgar um produto facilita sua aceitação.”, “O uso de personagens é um fator de estímulo de consumo, uma afirmação do subconsciente.”;
* **Dicas ao usuário (*Hints*) –** “Deveria ser na língua da pessoa que usa.”, ou ainda “As dicas sempre são fundamentais para facilitar o entendimento da ferramenta, e ser o primeiro *help* do usuário.”;
* **Capacidade de ser intuitiva –** “Todo sistema tem uma operacionalização intuitiva, por isso, facilita o interesse pelo uso.”, ou ainda “Me confundi na ferramenta em relação à ordem cronológica, até porque não costumamos ler e ainda existe a dificuldade do inglês.”; e
* **A ViXQueL –** “Tem boa intenção e finalidade, porém poderia ser mais clara em relação às suas funções. visual bom.”, ou ainda “Ótima ferramenta principalmente pela facilidade de operação”.

Então, analisando os comentários e notas dessa validação, é notória a aceitação da ViXQueL por parte dos usuários e sua facilidade na criação de consultas. A motivação no uso da ferramenta pelos usuários também foi verificada. E isso se deu por aprenderem um assunto que eles mesmos julgaram não terem condições de dominar ao verem a consulta textual XQuery gerada a partir da consulta visual criada por eles próprios.

Também é possível notar a convergência de opinião na maioria dos tópicos abordados, tais como o uso parametrizado das cores da ferramenta, o uso da internacionalização da ferramenta para o português, uma melhor disposição e exposição do nome da ferramenta ou ainda a falta de necessidade de mídias para atrair a atenção no uso da ferramenta.

No capítulo final, serão abordadas as discussões sobre os resultados obtidos, bem como possíveis trabalhos futuros e as dificuldades encontradas nesse trabalho.

# Trabalhos Futuros e Considerações Finais

Neste capítulo são abordadas as discussões finais deste trabalho, destacando as conclusões e principais contribuições, além de limitações e possíveis trabalhos futuros.

## Considerações Finais

O desenvolvimento deste trabalho foi principalmente motivado pela necessidade de colaboração remota quando se trata de obtenção de informação para tomada de decisão em uma corporação. Através do estudo de ferramentas criadas com propósitos semelhantes, foi possível identificar uma deficiência nesta questão.

Entretanto, atualmente com o crescente uso de XML, inclusive dos SGBD XML nativos, é importante a disponibilização de uma interface compatível com qualquer tipo de usuário, independente do seu nível de conhecimento. Deste modo, este trabalho apresentou também uma validação heurística qualitativa de ViXQueL, uma ferramenta web de criação de consultas visuais a SGBD XML nativos. Essa contribuição é válida para que em trabalhos futuros, melhoramentos e adição de novas funcionalidades sejam inseridas na ViXQueL, inclusive em uma possível expansão da expressão da ViXQueL em relação a XQuery, linguagem textual de consulta aos SGDB citados anteriormente.

As funcionalidades adicionadas a ViXQueL facilitam a colaboração na extração de informação de bancos de dados XML nativos. Essa colaboração, por sua vez, abrange tanto o nicho corporativo quanto o educacional. Permite-se inclusive, a colaboração voltada ao meio acadêmico visando a aprendizagem da linguagem textual de consulta XQuery.

A interface da ViXQueL evita que o usuário precise ter qualquer conhecimento desta linguagem para ter acesso às informações da base de dados, mas, caso desejado, o usuário pode visualizar a consulta textual em XQuery. Entretanto, é importante frisar que os elementos disponíveis apenas permitem que sejam criadas consultas utilizando um subconjunto da linguagem XQuery.

As principais contribuições deste trabalho foram a adição de componentes que permitem a colaboração na criação de consultas visuais em ViXQueL, uma ferramenta visual web para abstração de consultas XQuery e a validação heurística da atratividade e usabilidade da ferramenta ViXQueL.

## Trabalhos Futuros

Analisando o trabalho desenvolvido, é possível identificar algumas limitações que mesmo não afetando o resultado final esperado, seriam de grande valia se fossem implementadas. Algumas dessas limitações foram impostas por complexidade no desenvolvimento, outras por extrapolação do escopo, tornando-se inviável devido à restrição de tempo. Dentre estas limitações, algumas são complexas o suficiente para serem apontadas como possíveis trabalhos futuros.

Apesar do objetivo inicial deste trabalho ter sido atingido, é possível observar que ainda existem melhorias a se realizar. Considerando este aspecto, podem-se ressaltar alguns pontos que poderiam ser estendidos e melhorados.

Ao longo da apresentação deste trabalho foi possível notar claramente que o protótipo apresentado foi projetado para funcionar com o SGBD XML nativo Sedna (32) sendo assim, dependente de um SGBD XML nativo específico. Assim, sugere-se que o protótipo da ferramenta seja reprojetado para funcionar como uma interface, sendo facilmente adaptável a qualquer SGBD XML nativo. Para isto, deve-se realizar uma avaliação minuciosa da interface de acesso de cada SGBD XML nativo e então, remodelar o diagrama de classes do protótipo para atender às novas necessidades.

A expressividade de consultas da ferramenta pode ser melhorada, pois o protótipo atual contempla apenas um subconjunto da XQuery. Entretanto, deve-se levar em consideração a preocupação com a usabilidade, que foi existente no projeto das duas versões da ferramenta ViXQueL. Então, é importante que seja realizado um estudo aprofundado para determinar quais funcionalidades poderiam ser acrescidas à ferramenta sem prejudicar a usabilidade.

A adição de outros tipos de restrições e funções pré-definidas da XQuery, bem como a utilização do operador lógico OR são exemplos de funcionalidades que poderiam agregar valor à expressividade da ferramenta. Além destas, seria interessante permitir a criação de consultas com ordenação por mais de um elemento, visto que atualmente o protótipo dá suporte a consultas com ordenação por apenas um único elemento.

Com relação às funcionalidades existentes para gerenciamento de dados, seria interessante permitir a atualização e a remoção de dados, visto que a inserção e consulta já estão disponíveis. Assim, a ferramenta forneceria o ciclo completo de funcionalidades para o gerenciamento de dados.

Outra melhoria seria a adaptação do retorno da consulta de acordo com as preferências do usuário, permitindo visualizar o retorno da consulta no formato padrão, em XML puro, ou ainda, moldar o retorno utilizando um arquivo XSLT, permitindo assim um vasto conjunto de tipos de retorno.

Pode-se citar também como uma contribuição importante, a criação de um modelo que permita cada usuário efetuar o *seu cadatro* com login e uma senha, e ter todo seu histórico de atividades salvo na ferramenta ViXQueL.

Por fim, algumas melhorias para a interface também poderiam ser realizadas. Um exemplo é a permissão de exclusão das restrições das entidades sem a necessidade de excluir toda a entidade. Um segundo exemplo é a adição de funcionalidades para a área de construção de consultas, tais como, aumentar e diminuir o zoom. Mais um exemplo é a tradução completa para a língua portuguesa. Ou ainda, cria um logo tipo com o nome da ferramenta, a fim de cativar e excitar o usuário a interagir com a ferramenta. Outra sugestão é adaptar a ferramenta para trabalhar com outros *browsers*, como o Internet Explorer (68), pois atualmente o protótipo da ferramenta apenas permite a correta visualização no Mozilla Firefox (69).

# Referências

1. **W3C.** Extensible Markup Language (XML). *Extensible Markup Language (XML).* [Online] W3C. [Cited: 05 20, 2010.] http://www.w3.org/XML/.

2. **Fiebig, Thorsten, et al.** Anatomy of a native xml base management system. *The VLDB Journal.* 12 2002, Vol. 11, 4, pp. 292–314.

3. **W3C.** Xquery 1.0: An xml query language. *An xml query language.* [Online] W3C. [Cited: 06 10, 2010.] http://www.w3.org/TR/xquery/.

4. **Bastos, Petrus.** ViXQueL: uma ferramenta visual Web para acesso a banco de dados XML nativo. 02 2009.

5. **Cruz, Isabel F., Mendelzon, Alberto O. and Wood, Peter T.** A graphical query language supporting recursion. *ACM SIGMOD Record.* 1987, pp. 323–330.

6. —. Recursive queries without recursion. *Expert Database Conference.* 1988, pp. 645-666.

7. **Consens, Mariano P. and Mendelzon., Alberto O.** The g+/graphlog visual query system. *ACM SIGMOD Record.* 1990, p. 388.

8. **Paredaens, Jan, et al.** An overview of good. *ACM SIGMOD Record.* 1992, pp. 25–31.

9. **Paredaens, Jan, Peelman, Peter and Tanca, Letizia.** G-log: A graph-based query language. *IEEE Transactions on Knowledge and Data Engineering.* 1995, pp. 436–453.

10. **Comai, Sara, et al.** A schema-based approach to modeling and querying www data. *In Proceedings of the Third International Conference on Flexible Query Answering Systems (FQAS).* 1998, pp. 110–125.

11. **Comai, Sara, Damiani, Ernesto and Fraternali, Piero.** Computing graphical queries over xml data. *ACM Transactions on Information Systems.* 2001, pp. 371–430.

12. **Comai, Sara.** Graph-based guis for querying xml data: the xml-gl experience. *Proceedings of the 2001 ACM symposium on Applied computing.* 2001, pp. 269–274.

13. **Braga, Daniele, Campi, Alessandro and Ceri, Stefano.** Xqbe (xquery by example): A visual interface to the standard xml query language. *ACM Transactions on Database Systems.* 2005, pp. 398–443.

14. **Zloof., Moshé M.** Query-by-example: A data base language. *IBM Systems Journal.* 1977, pp. 324–343.

15. **Qin, Zheng, et al.** A graphical xquery language using nested windows. *In Proceedings of 5th International Conference on Web Information System Engineering.* November 2004, pp. 681–687.

16. **Munroe, Kevin D. and Papakonstantinou, Yannis.** Bbq: A visual interface for integrated browsing and querying of xml. *In Proceedings of the Fifth Working Conference on Visual Database Systems (VDB 5).* 2000, pp. 277–296.

17. Microsoft windows. [Online] Microsoft. [Cited: 06 10, 2010.] http://www.microsoft.com/windows/.

18. **Baru, Chaitanya K., et al.** Features and requirements for an xml view definition language: Lessons from xml information mediation. *In The Query Languages Workshop.* 1998.

19. **W3C.** Document type definition (dtd). *DTD - Document type definition.* [Online] W3C. [Cited: 06 10, 2010.] http://www.w3.org/TR/REC-xml/#dt-doctype.

20. **Karam, Marcel, et al.** Xqueryviz: A visual dataflow xquery tool. *In Proceedings of the Advanced International Conference on Telecommunications and International Conference on Internet and Web Applications and Services (AICT-ICIW ’06).* 2006, p. 196.

21. Xml path language (xpath). [Online] W3C. [Cited: 06 10, 2010.] http://www.w3.org/TR/xpath/.

22. **W3C.** XML Schema Part 0: Primer Second Edition. *XML Schema.* [Online] W3C. [Cited: 06 10, 2010.] http://www.w3.org/TR/xmlschema-0/.

23. **Li, Xiang, Gennari, John H. and Brinkley, James F.** Xgi: a graphical interface for xquery creation. *In Proceedings of the 2007 American Medical Informatics Association Annual Symposium (AMIA).* 2007, pp. 453–457.

24. **Garshol, Lars Marius.** BNF and EBNF: What are they and how do they work? [Online] [Cited: 06 10, 2010.] http://www.garshol.priv.no/download/text/bnf.html.

25. Apache Tomcat. [Online] The Apache Software Foundation. [Cited: 06 10, 2010.] http://tomcat.apache.org/.

26. About JavaScript. [Online] Mozilla. [Cited: 06 10, 2010.] http://developer.mozilla.org/en/docs/About\_JavaScript.

27. XHTML2 Working Group Home Page. [Online] W3C. [Cited: 06 10, 2010.] http://www.w3.org/MarkUp/.

28. Cascading Style Sheets Home Page. [Online] W3C. [Cited: 06 10, 2010.] http://www.w3.org/Style/CSS/.

29. **Fuchs, Thomas.** script.aculo.us. [Online] [Cited: 06 10, 2010.] http://script.aculo.us/.

30. **Gruhier, Sébastien.** Prototype Graphic Framework. [Online] [Cited: 06 10, 2010.] http://prototype-graphic.xilinus.com/.

31. jscanvas - a javascript graphic library. [Online] [Cited: 06 10, 2010.] http://code.google.com/p/jscanvas/.

32. SEDNA - Native XML Database System. [Online] Institute for System Programming RAS. [Cited: 06 10, 2010.] http://modis.ispras.ru/sedna/index.htm.

33. AJAX Tutorial. [Online] W3Schools. [Cited: 06 10, 2010.] http://www.w3schools.com/Ajax/Default.Asp.

34. OMG's MetaObject Facility. [Online] Object Management Group, Inc. [Cited: 06 10, 2010.] http://www.omg.org/mof/.

35. Eclipse Modeling Framework Project (EMF). [Online] The Eclipse Foundation. [Cited: 06 10, 2010.] http://www.eclipse.org/modeling/emf/.

36. Eclipse.org Home. [Online] The Eclipse Foundation. [Cited: 06 10, 2010.] http://www.eclipse.org/.

37. Kermeta - Breathe life into your metamodels. [Online] IRISA. [Cited: 06 10, 2010.] http://www.kermeta.org/.

38. ATL - a model transformation technology. [Online] The Eclipse Foundation. [Cited: 06 10, 2010.] http://www.eclipse.org/m2m/atl/.

39. Eiffel Software . [Online] Eiffel. [Cited: 06 10, 2010.] http://www.eiffel.com/.

40. Java.sun.com - The Source for Java Developers. [Online] [Cited: 06 10, 2010.] http://java.sun.com/The Source for Java Developers.

41. **Schaffers, Hans, et al.** *The Future Workspace - Perspectives on Mobile and Collaborative Working.* s.l. : Telematica Instituut, The Netherlands, www.telin.nl, 2005. p. 112. http://www.ami-communities.eu/pub/bscw.cgi/d163187/The%20Future%20Workspace.pdf. ISBN-10: 90-75176-00-7 | ISBN-13: 978-90-75176-00-1.

42. *SOFTWARE CATCHES THE TEAM SPIRIT New computer programs may soon change the way groups of people work together -- and start delivering the long-awaited payoff from office automation.* **Richman, Louis S. and Slovak, Julianne.** 06 08, 1987, FORTUNE. http://money.cnn.com/magazines/fortune/fortune\_archive/1987/06/08/69109/index.htm.

43. IBM Collaboration Software - Lotus software. [Online] [Cited: 06 10, 2010.] http://www-01.ibm.com/software/lotus/.

44. Software de colaboração - Conecta e integra pessoas às informações de negócios. *IBM Software de colaboração - Brasil.* [Online] IBM. [Cited: 06 13, 2010.] http://www-01.ibm.com/software/br/info/itsolutions/collaboration/.

45. **Media, Master New.** Online Collaboration: Novas Aplicações E Serviços Web. *Online Collaboration: Novas Aplicações E Serviços Web - Sharewood Picnic.* [Online] masternewmedia. [Cited: 06 28, 2010.] http://www.masternewmedia.org/pt/colaboracao\_on\_line/aplicacoes-web/online-collaboration-novas-aplicacoes-e-servicos-web-sharewood-picnic-20080114.htm.

46. **Wikipedia.** Wiki. [Online] Wikipedia. [Cited: 06 13, 2010.] http://pt.wikipedia.org/wiki/Wiki.

47. **Wallingford, Ted.** *Switching to VoIP.* Sebastopol : O`Reilly, 2005. ISBN-10: 0-596-00868-6 | ISBN-13: 978-0-596-000868-0.

48. Design - Wikipédia. *Design.* [Online] Wikipédia. [Cited: 06 18, 2010.] http://pt.wikipedia.org/wiki/Design.

49. **Newman, Mark W. and Landay, James A.** Sitemaps, Storyboards, and Specifications: A Sketch of Web Site Design Practice as Manifested Through Artifacts. Agosto 2000, pp. 263-274. {newman, landay}@cs.berkeley.edu.

50. **Ivory, Melody Y. and Hearst, Marti A.** Usability and the Web - Improving Web Site Design. *IEEE INTERNET COMPUTING.* 03 2002, pp. 56-63. http://computer.org/internet/.

51. **Sutcliffe, Alistair G.** Assessing the Reliability of Heuristic Evaluation for Website Attractiveness and Usability. 2002, pp. 183-198.

52. **ISO.** ISO 9241: Ergonomic requirements for office systems with visual display terminals (VDTs). *International Standards Organisation.* 1997.

53. **Nielsen, Jakob.** *Usability engineering.* 1993. p. 362. ISBN-10: 0125184069 | ISBN-13: 978-0125184069.

54. **Nielsen, Jacob.** *Designing Web Usability: The Practice of Simplicity.* Thousand Oaks : New Riders Publishing, 1999. p. 419. ISBN:156205810X.

55. **Nielsen, Jacob and Molich, Rolf.** Heuristic evaluation of user interfaces. *Proceedings of the SIGCHI conference on Human factors in computing systems: Empowering people.* 1990, pp. 249-256.

56. **Sutcliffe, Alistair, et al.** Model mismatch analysis: towards a deeper explanation of users' usability problems. 2000, Vol. 19, pp. 43-55.

57. **Kristof, Ray and Satran, Amy.** *Interactivity by design: creating and communicating with new media.* Mountain View, CA : Adobe Press, 1995. p. 144. ISBN-10: 1568302215 | ISBN-13: 978-1568302218.

58. **Mullet, Kevin and Sano, Darrell.** *Designing visual interfaces: communication oriented techniques.* Englewood Cliffs NJ : unSoft Press, 1995. p. 304. ISBN-10: 0133033899 | ISBN-13: 978-0133033892.

59. **Spool, Jared M., et al.** *Web site usability: a designer's guide.* San Francisco, CA : s.n., 1998. p. 176. ISBN-10: 155860569X | ISBN-13: 978-1558605695.

60. **Sutcliffe, Alistair G.** *Multimedia and virtual reality: designing multisensory user interfaces.* Mahwah, NJ : s.n., 2001. p. 352. ISBN-10: 080583950X | ISBN-13: 978-0805839500.

61. —. User-centred design for multimedia applications. *Proceedings ICMCS: IEEE Conference on Multimedia Computing and Systems.* 06 7-11, 1999, Vol. 1, pp. 116-123.

62. **Maslow, Abraham H.** *Motivation and personality.* 3. New York : s.n., 1987. p. 293. ISBN-10: 0060419873 | ISBN-13: 978-0060419875.

63. **Fogg, B. J.** Persuasive computer: perspectives and research directions. *Proceedings: Human Factors in Computing Systems.* Abril 18-23, 1998, pp. 225-232.

64. **IBM.** Ease of use: design principles. *Design principles.* [Online] IBM. [Cited: 06 05, 2010.] http://www-01.ibm.com/software/ucd/designconcepts/designbasics.html.

65. **Teichmann, Karin and Zins, Andreas H.** Information Elements on DMO-Websites: Alternative Approaches for Measuring Perceived Utility. *Information and Communication Technologies in Tourism 2008.* s.l. : Springer Vienna, 2008, pp. 209-219.

66. **Google.** Google Docs. *Google Docs.* [Online] Google. [Cited: 05 12, 2010.] http://www.google.com/google-d-s/intl/pt-BR/tour1.html.

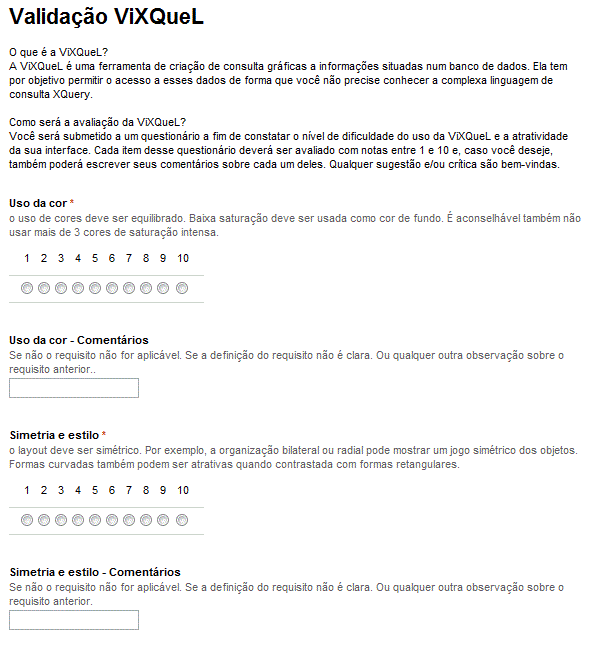
67. —. Orkut. *Orkut.com.* [Online] Google. [Cited: 06 29, 2010.] www.orkut.com.

68. **Corporation, Microsoft.** Internet Explorer 8. *Internet Explorer 8.* [Online] Microsoft Corporation. [Cited: 05 30, 2010.] http://www.microsoft.com/brasil/windows/internet-explorer/.

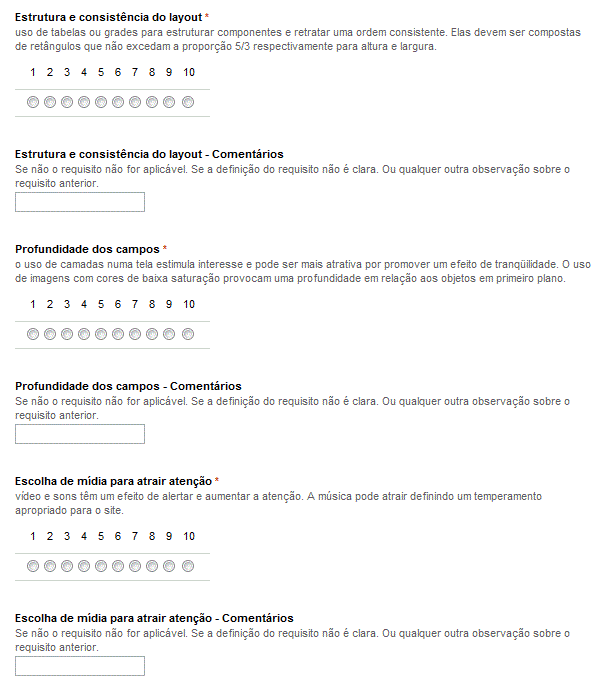
69. **Foundation, Mozilla.** Firefox. *Firefox.* [Online] Mozilla Foundation. [Cited: 05 30, 2010.] http://br.mozdev.org/.

70. **Carstensen, Peter H. and Schmidt, Kjeld.** Computer supported cooperative work: New challenges to systems design. *Handbook of Human Factors/Ergonomics.* 2003, pp. 619-636.

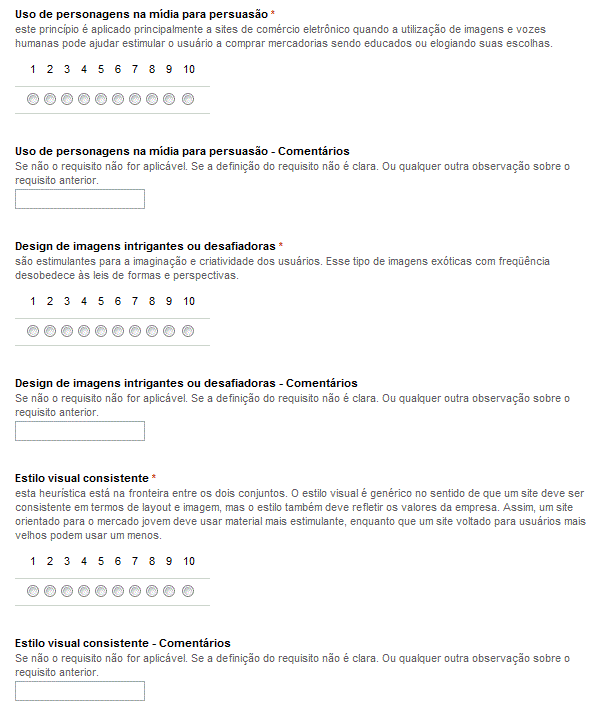
Anexo I – parte I



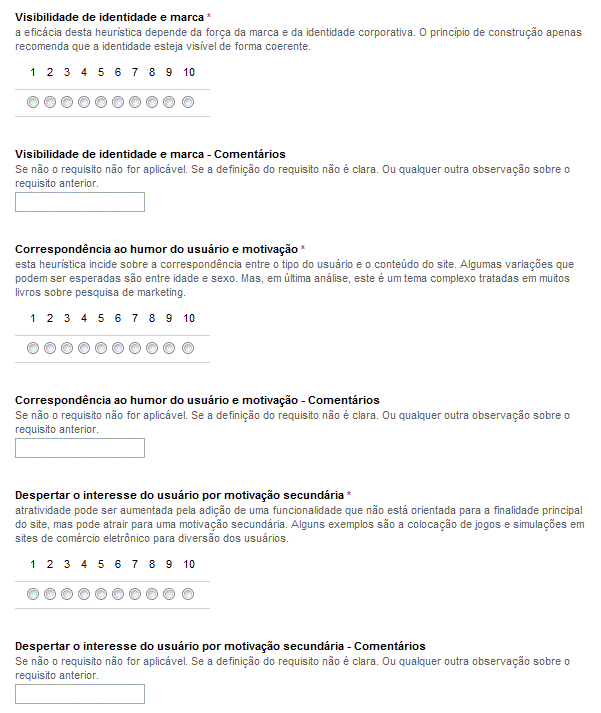
Anexo I – parte II



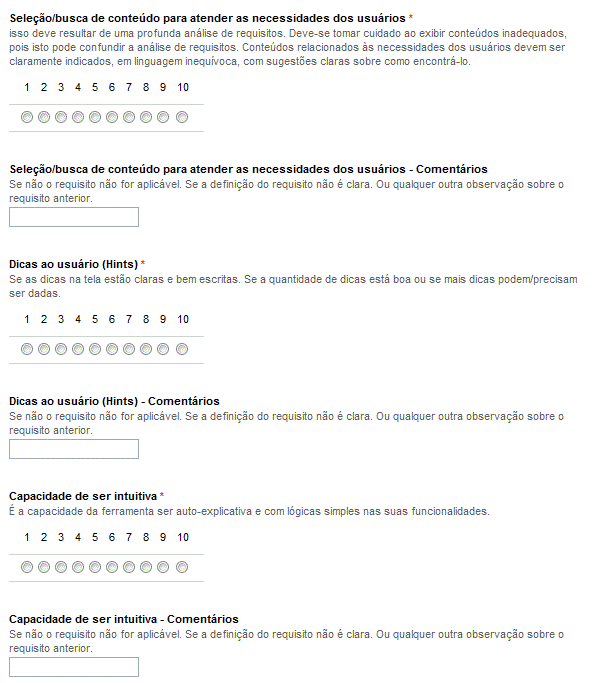
Anexo I – parte III



Anexo I – parte IV



Anexo I – parte V



Anexo I – parte VI

