UNIVERSIDADE FEDERAL DA PARAÍBA
CENTRO DE CIÊNCIAS EXATAS E DA NATUREZA
DEPARTAMENTO DE INFORMÁTICA
PROGRAMA DE PÓS-GRADUAÇÃO EM INFORMÁTICA

STYLO: Um Framework Voltado para o Desenvolvimento de Aplicações Baseadas em Vídeo Digital

JULIO CÉSAR FERREIRA DA SILVA

JOÃO PESSOA - PB
NOVEMBRO DE 2011

JULIO CÉSAR FERREIRA DA SILVA

STYLO: Um Framework Voltado para o Desenvolvimento de Aplicações Baseadas em Vídeo Digital

Dissertação submetida ao Programa de Pós-Graduação em Informática da Universidade Federal da Paraíba como parte dos requisitos necessários para obtenção do título de Mestre em Informática.

Orientador: Profa. Dra. Tatiana Aires Tavares

JOÃO PESSOA - PB
NOVEMBRO DE 2011

AGRADECIMENTOS

A Deus, por sempre me guiar nas escolhas que fiz e que me permitiram chegar até aqui.

À minha mãe, Iêda Angelina da Silva, que consegue ser uma grande mãe e uma grande amiga em todos os momentos da minha vida e sempre me incentivou a continuar os estudos mostrando que esse era o único bem impossível de ser tomado.

Ao meu Tio, Carlos Alberto Faustino Silva, que da infância até hoje me mostra que o melhor caminho para o sucesso é o estudo.

À minha avó, Dona Nem, que infelizmente nos deixou no meio do meu curso e sei que ela estaria muito feliz por mais essa vitória.

À minha querida irmã, Juliete Christine Ferreira da Silva, que, apesar dos aperreios que me faz, nunca deixou de me admirar e me incentivar.

Enfim, à minha família, que sempre me deu o suporte necessário para todas as conquistas, me apoiando nas decisões, acompanhando meu crescimento e ajudando nos momentos de necessidade.

À minha orientadora e amiga, Dra. Tatiana Aires Tavares, que me acolheu como uma mãe no laboratório e que sempre confiou em mim.

Aos meus companheiros de laboratório, em especial Marcello Galdino Passos, Elenilson Vieira, Erick Melo e Anderson Vinícius, que fizeram contribuições significativas para a conclusão deste trabalho.

Por fim, aos amigos que estiveram comigo durante todos esses anos, aos que ainda estão presentes e aos que já não participam mais da minha vida, muito obrigado por tudo.

Aos professores que passaram por mim durante toda a minha vida acadêmica e, também, em minha vida escolar.

À Capes que proporcionou o auxílio financeiro necessário, sem o qual este trabalho não poderia ter sido realizado.

Divido essa conquista com todos que participaram da minha formação acadêmica direta e indiretamente. Muito obrigado.

Se as coisas são inatingíveis... ora! Não é motivo para não querê-las.
Que tristes os caminhos, se não fora a mágica presença das estrelas.
(Mário Quintana)

RESUMO

O progresso da Ciência e Tecnologia aliado à melhoria das redes de comunicação, favorecem o surgimento de serviços para lidar com volumes crescentes de dados e com o alto poder de transmissão disponível. Aplicações baseadas em vídeo digital estão cada vez mais populares e atingem os mais diversos campos: Telemedicina, Arte e Tecnologia, entretenimento, publicidade, dentre outros. Proporcionalmente, são exigidos níveis cada vez mais altos de qualidade e prazos de entrega cada vez mais curtos para o desenvolvimento de sistemas de software. Diante disso, este trabalho faz um mapeamento do estado da arte no processo de desenvolvimento de aplicações baseadas em vídeo digital e mostra a necessidade de se ter um framework que forneça uma abstração para o desenvolvimento de sistemas nessa área. Além disso, é apresentado o STYLO, um arcabouço de software que tem como objetivo encapsular experiências adquiridas no desenvolvimento de tecnologias desse âmbito, provendo para os programadores formas de reuso, usabilidade e padronização de código e trazendo maior produtividade em suas implementações.

Palavras-chave: Frameworks, Codificação, Transmissão, Vídeo.

ABSTRACT

The progress of science and technology coupled with the improvement of communication networks favoring the emergence of services to handle with growing volumes of data and the high transmission power available. Applications based on digital video are increasingly popular. They are widespread in multiples fields: telemedicine, artistic performances, techno-scientific events, entertainment, and advertising, among others. Proportionately, they are increasingly required higher levels of quality and delivery times getting shorter for the development of software systems. Therefore, this paper aims to map the state of the art in the development of applications based on digital video and show the necessity of having a framework that provides an abstraction for the development of systems in this area. Finally, is presented a case study of STYLO, a software framework that aims to encapsulate experiences in developing technologies that scope, providing ways for developers to reuse, usability and standardization of code and bringing greater productivity in their implementations.

Key-words: Frameworks, Encoding, Streaming, Video.

Lista de Abreviaturas

AAC – Advanced Audio Coding

ARPANet – Advanced Research Projects Agency Network

API – Application Programming Interface

AVI – Audio Video Interleave

CNCTI – Conferência Nacional de Ciência, Tecnologia e Inovação

EAT – Experiments in Art and Technology

GTAVCS – Grupo de Trabalho Ambiente de Videocolaboração em Saúde

GTMDA – Grupo de Trabalho de Mídias Digitais e Arte

HTTP – Hipertext Terminal Protocol

HD – High Definition

IHC – Interação Humano Computador

IP – Internet Protocol

JPEG – Joint Photographic Experts Group

JNI – Java Native Interface

JVM – Java Virtual Machine

LAN – Local Area Network

LAViD – Laboratório de Aplicações de Vídeo Digital

LASID – Laboratório de Sistemas Digitais

LAT – Laboratório de Telemedicina

MAN – MetropolitanArea Network

MCU – Multipoint Control Unit

MPEG-2 – Moving Picture Experts Group 2

MPEG-4 - Moving Picture Experts Group 4

MP3 – MPEG-1/2 Audio Layer 3

ORB – Object Request Broker

PNG – Portable Network Graphics

QoS – Quality of Service

RMI – Remote Method Invocation

RNP – Rede Nacional de Ensino e Pesquisa

RTP – Real-Time TransportProtocol

SD – Standard Definition

SLTV – Streaming Livre TV

SNMP – Simple Network Management Protocol

TCP – TransmissionControlProtocol

TIC – Tecnologias da Informação e Comunicação

TS – Transport Stream

UDP – UserDatagramProtocol

UML – Unified Modeling Language

UFPB – Universidade Federal da Paraíba

UNIFESP – Universidade do Estado de São Paulo

WAN – WideArea Network

WRNP – Workshop da RNP

WMV – Windows Media Video

Sumário

Capítulo

1

Introdução

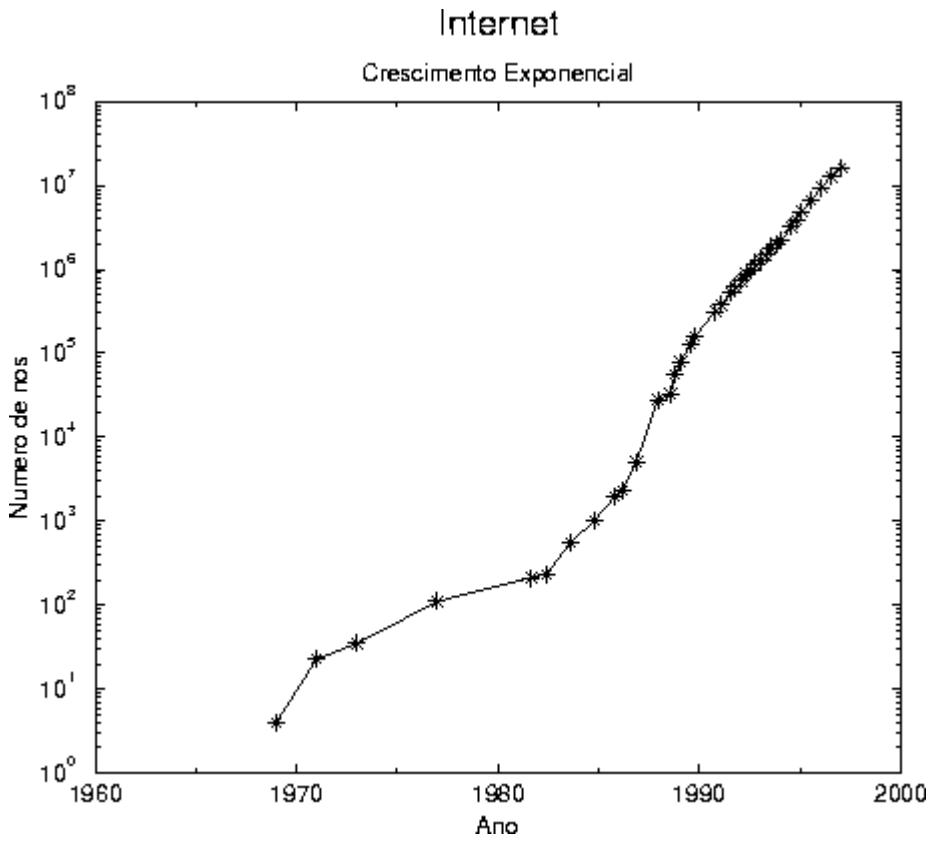
Existe uma tendência cada vez mais acentuada de adoção das tecnologias de informação e comunicação em escolas, empresas de diversas áreas e, sobretudo com a disseminação dos aparelhos digitais, no cotidiano contemporâneo.

O desenvolvimento das tecnologias de redes de computadores tem se dado em ritmo acelerado nos últimos trinta anos. Tal desenvolvimento iniciou-se com um experimento acadêmico, o ARPANet em 1969 e, desde então, surgiram várias topologias e tecnologias que deram origem às redes de diferentes portes (LANs, MANs e WANs) e diferentes arquiteturas. Evoluções mais recentes em relação ao meio físico para a transmissão digital (fibras óticas) e equipamentos de conexão de rede computadorizados, criaram subsídios para novas técnicas de transmissão que apresentam um desempenho muito melhor, maximizando a velocidade e minimizando as perdas de dados durante a transmissão. Essas novas tecnologias deram origem às redes de alta velocidade ou, como algumas vezes são chamadas, redes Gigabits e nos últimos três anos, as redes Terabits (SILVEIRA).

O uso de sistemas multimídia tem tomado uma grande abrangência no que se refere a possíveis contextos de uso. Exemplos desses contextos são: educação à distância, telemedicina, espetáculos e eventos artístico-tecnológicos e videoconferência. Soluções com funcionalidades multimídia avançadas estão se tornando cada vez mais frequentes e o volume de mídia digital produzida tanto para uso profissional como para uso pessoal cresce constantemente. Todos esses "provedores de conteúdo" compartilham as mesmas preocupações: gestão de conteúdo, capacidades de rede e dispositivos, proteção da informação contra acesso não autorizado, formatos de dados. Essas soluções impulsionam novos paradigmas de comunicação, novas expressões artísticas, novas maneiras de realizar trabalhos operacionais e, consequentemente, passam a fazer parte do cotidiano da sociedade.

As mudanças em curso podem gerar impactos e efeitos tanto nas sociedades como na economia mundial, já que a difusão das novas tecnologias acontece em escala global (PALHARES 2005). A Figura 1 ilustra o crescimento dos nós na rede durante os anos e mostra que cada vez mais se tem suporte para serviços baseados em vídeo digital em larga escala.

Figura 1. Crescimento dos nós de rede durante os anos.



Fonte: (PATTERSON 1995)

O poder computacional disponível em nossos dias permite explorar ainda mais sistemas multimídia. As novas tecnologias, relacionadas a uma revolução informacional, oferecem uma infraestrutra comunicacional que permite a interação em rede de seus integrantes. Essa comunicação de duas vias entre usuário e o software é chamada de interatividade. O conteúdo multimídia também não precisa ser necessariamente pré-processado ou estar armazenado em memória para que seja reproduzido ou retransmitido. Um espectador conectado ao sistema por uma rede tem acesso ao conteúdo capturado por câmeras de vídeo ou outros dispositivos em tempo real. Essa facilidade no acesso a tal conteúdo faz com que a procura por esses serviços cresça. Isso pode ser ilustrado na Figura 2, que mostra a

porcentagem de expectadores online nos Estados Unidos por grupo de idade entre 2008 e 2011 e faz uma projeção até 2014.

Figura 2. Indivíduos que assistem conteúdo de vídeo digital online pelo menos uma vez por mês.

	2008	2009	2010	2011	2012	2013	2014
0-11	50.0%	54.1%	58.1%	61.9%	67.0%	70.8%	75.1%
12-17	70.0%	74.9%	79.0%	82.8%	87.1%	90.1%	91.8%
18-24	80.1%	83.2%	86.0%	90.1%	93.0%	94.1%	95.2%
25-34	75.1%	80.0%	84.1%	88.0%	91.1%	93.1%	93.9%
35-44	69.1%	74.0%	77.1%	79.9%	82.9%	85.9%	88.0%
45-54	50.2%	54.9%	58.1%	60.9%	64.0%	66.1%	68.1%
55-64	35.1%	40.2%	43.8%	49.1%	53.1%	55.9%	59.0%
65+	18.8%	22.7%	25.8%	28.8%	33.2%	36.2%	39.0%

Fonte: eMarketer, Abril 2010.

O desenvolvimento de aplicações de software envolvendo vídeo digital é também afetado por essas mudanças, uma vez que se tem aumento da demanda e redução do time-to-market para produtos relacionados à área. A presença de uma camada de abstração que incorpore funções recorrentes e contemple aplicações multimídia independentemente do contexto que estão sendo inseridas, agrega produtividade e confiabilidade no processo de desenvolvimento de software. Frameworks fornecem um nível de abstração apropriado para alcançar este objetivo.

Nesse cenário de convergência digital, uso plural dos artefatos computacionais e crescimento das informações baseadas em vídeo digital é que a proposta deste trabalho é fundamentada. Em resposta aos anseios cada vez mais urgentes, pois soluções personalizáveis, adaptáveis e rápidas, o framework STYLO é apresentado. Nas subseções seguintes são descritos os objetivos deste trabalho, alguns trabalhos desenvolvidos no laboratório de aplicações de vídeo digital (LAViD), o problema que foi identificado para a concepção deste trabalho, bem como suas contribuições. Por fim, é apresentada a estrutura da dissertação.

1.1 Objetivos

Este trabalho tem como *objetivo principal* o desenvolvimento de um framework que dê suporte ao desenvolvimento de aplicações baseadas em vídeo digital. Esse framework possibilitará o reuso de código, acelerando o desenvolvimento de tais sistemas, possibilitará um bom nível de abstração, suporte para captura, exibição, codificação, transmissão e gerenciamento de fluxos de mídia e suporte para medição e monitoramento da rede. Além disso, facilitará a incorporação de funcionalidades que não estão relacionadas propriamente a multimídia, porém servem de complemento para aplicações com esse fim, como segurança e criptografia de dados, mapa geográfico, temporizador de operações, customizador de interface gráfica, dentre outros.

Para alcançar o objetivo principal supracitado, têm-se os seguintes *objetivos específicos*:

- Levantamento do estado da arte e investigação de trabalhos relacionados.
- Estudo de conceitos acerca de vídeo digital e de frameworks.
- Proposição, especificação e desenvolvimento de um framework para o desenvolvimento de aplicações baseadas em vídeo digital.
- Verificação do framework proposto em termos de funcionalidades oferecidas através do desenvolvimento de aplicações teste.
- Validação o framework proposto através do seu uso supervisionado por programadores.

1.2 Trabalhos Correlatos na Instituição

As pesquisas desenvolvidas pelo Laboratório de Aplicações e Vídeo Digital (LAVID) da Universidade Federal da Paraíba são realizadas em parceria com outras universidades, institutos de pesquisa e empresas da iniciativa privada. Tais pesquisas foram extremamente importantes para a construção de uma vasta experiência no desenvolvimento de aplicações voltadas para vídeo digital. A seguir, é feita uma descrição dos projetos que ilustram essa experiência.

O Grupo de Trabalho de Vídeo Digital (GT VD) teve por objetivo implantar uma infraestrutura baseada na RNP que ofereça suporte a aplicações envolvendo manipulação de vídeo digital. Coube ao GT VD incentivar e fornecer condições para criação, armazenamento e transmissão de conteúdo na forma de vídeo digital no país.

O Grupo de Trabalho de Televisão (GT TV) surgiu como desdobramento dos projetos I2TV (Infra-estrutura Internet2 para Desenvolvimento e Teste de Programas e Ferramentas para TV Interativa), HiTV (Desenvolvimento de Software e Hardware para Sistemas de Televisão Digital de Alta Definição) e dos GTs de Vídeo Digital da RNP. O projeto desenvolveu uma plataforma que facilita o acesso de usuários de características heterogêneas ao conteúdo de canais de TV distribuídos através da Internet, utilizando uma infraestrutura IPTV.

O GigaVR teve como objetivo o desenvolvimento e demonstrações de uma plataforma voltada para a criação de aplicações da realidade virtual imersiva sob redes de altíssima velocidade. Nele foi apresentada a teleoperação de um robô associada à transmissão de vários fluxos de vídeos sincronizados com alta qualidade e em tempo real, com a exibição desses vídeos em uma Caverna Digital.

O Grupo de Trabalho de Mídias Digitais e Arte (GT MDA) teve como foco principal oferecer formas mais avançadas para Interação Humano Computador (IHC), as quais permitam o entrelaçamento de agentes humanos e sintéticos em espaços midiáticos compartilhados e distribuídos, em tempo real, através de redes de computadores de alta velocidade e com grande volume de informação. A ideia central é a construção de uma ferramenta de gerenciamento que concentre ao máximo o controle de todos os dispositivos de hardware e software envolvidos para execução de evento de cunho artístico-tecnológico. O esforço do GTMDA resultou na criação da Arthron - uma ferramenta para facilitar a execução de performances artísticas que utilizam representações midiáticas e o compartilhamento de espaços reais e virtuais em tempo-real. A Arthron tem por principal funcionalidade oferecer ao usuário uma interface simples para manipulação de diferentes fontes/fluxos de mídia simultâneos. Dessa forma, o usuário pode, remotamente, adicionar, remover, configurar o formato de apresentação e programar a exibição no tempo (quando apresentar) e no espaço (onde apresentar) dos fluxos de mídia de um espetáculo. A ferramenta é composta por quatro componentes principais: Articulador (*Manager*), que dentre outras a principal função é efetuar o gerenciamento dos fluxos de vídeo; Agentes Codificadores (*Encoders*), responsáveis por capturar conteúdo multimídia em tempo real ou de arquivos, e transmiti-lo; Agentes Decodificadores (*Decoders*), que recebem em reproduzem conteúdo multimídia recebidos; Refletores (*Reflector*), componentes que replicam um único fluxo de mídia para N destinos espalhados pela rede; Servidores de Vídeo (*VideoServer*), capazes de publicar vídeos na web; Gerente de Mapa (*Map Manager*) que

fornecem uma apresentação gráfica em mapas da localização dos componentes conectados ao sistema; e Criador Cenário (*Scenario Maker*), com o qual é possível automatizar a troca de fluxos. Essas mídias podem ser enviadas em alta, média e baixa definição, simultaneamente, tanto para decodificadores específicos na rede quanto para a Internet (MELO 2010).

O Grupo de Trabalho em Ambientes de Vídeo Colaboração em Saúde (GT AVCS) é uma iniciativa conjunta dos laboratórios LAVID, LASID (Laboratório de Sistemas Digitais) e LARQSS (Laboratório de Arquitetura e Sistemas de Software) da UFPB e LAT (Laboratório de Telemedicina) da UNIFESP. Propõe uma infraestrutura de hardware e software com gerência remota para captura e distribuição segura de múltiplos fluxos simultâneos a fim de prover suporte a diversos cenários de vídeo colaboração em saúde. A ferramenta desenvolvida pelo GTAVCS é baseada na Arthron, porém acrescenta algumas funcionalidades para se adequar ao contexto da telemedicina. Favorecendo a transmissão de cirurgias, a ferramenta conta com um componente adicional capaz de gerar um fluxo de áudio e vídeo e receber N fluxos - um em alta qualidade e os demais em baixa. O usuário do componente tem a autonomia de escolher dentre os vídeos que ele está recebendo, qual chegará em alta qualidade. O sistema também é capaz de capturar o vídeo de diferentes dispositivos no ambiente cirúrgico, a exemplo de câmera do foco, endocâmera, câmera IP e estabelecer a comunicação bidirecional entre o médico que realiza a cirurgia e os espectadores remotos. É provida também uma infraestrutura de segurança da informação que inclui autenticação de usuários e criptografia dos dados trafegados na rede.

1.3 Identificação do problema

O desenvolvimento de ferramentas, como as legadas pelos grupos de trabalho GT VD e GT MDA, mostra que a necessidade de determinados componentes de software é recorrente. Uma vez que não exista modularização e reaproveitamento de componentes, um alto e desnecessário nível de retrabalho é demandado. Players, codificadores e streamers de mídia, por exemplo, são refeitos e tendo que passar pelas mesmas dificuldades e riscos. Surge, a partir daí, a necessidade de uma abstração de software como um arcabouço que reúna e encapsule funcionalidades comuns a aplicações dentro do escopo de vídeo digital.

Segundo Appleton (APPLETON), um framework é definido como uma arquitetura reusável que fornece comportamento e estrutura genérica para uma família de abstrações de software. Neste trabalho é adotada a definição de Fayad e Johnson (FAYAD e JOHNSON 2000) no qual um framework é definido como uma aplicação semi-completa contendo

componentes estáticos e dinâmicos os quais podem ser adaptados para produzir aplicações específicas dentro de um domínio. Diante disso, percebeu-se a necessidade de um processo sistêmico para o provimento do arcabouço ferramental imprescindível para apoiar o desenvolvimento de aplicações baseadas em vídeo digital.

1.4 Contribuições

Neste trabalho elencamos como principal contribuição teórica a proposição de uma arquitetura genérica para aplicações baseadas em vídeo digital instanciada através de um framework, o STYLO. Dessa forma, o processo de concepção dessa categoria de aplicações é simplificado e poder-se-á adicionar personalizações e adaptações para tornar a arquitetura flexível.

Por outro lado a instanciação do modelo arquitetural proposto gerou outras contribuições práticas deste trabalho, são elas:

- desenvolvimento de um framework de implementação voltado a aplicações baseadas em vídeo digital;
- construção de uma aplicação no contexto de telemedicina utilizando o framework proposto;
- utilização da aplicação construída em transmissões de cirurgias ao vivo, treinamento e validação do framework proposto por desenvolvedores.

1.5 Estrutura da Dissertação

Esta dissertação está organizada em sete capítulos. O primeiro capítulo é este que se encerra. O segundo capítulo apresenta a fundamentação teórica necessária ao desenvolvimento do trabalho, abordando aspectos inerentes a vídeo digital e de frameworks. O terceiro capítulo descreve os trabalhos correlatos encontrados e uma análise dos mesmos. O quarto capítulo apresenta a API, a arquitetura, a visão tecnológica e outros aspectos do framework STYLO. O quinto capítulo apresenta os resultados obtidos, como aplicações geradas que verificam as funcionalidades do framework. O sexto capítulo apresenta testes realizados com o framework, apresentando a metodologia utilizada e os resultados obtidos. O último capítulo apresenta as considerações finais e propõe os próximos passos da pesquisa.

Capítulo

2

Fundamentação Teórica

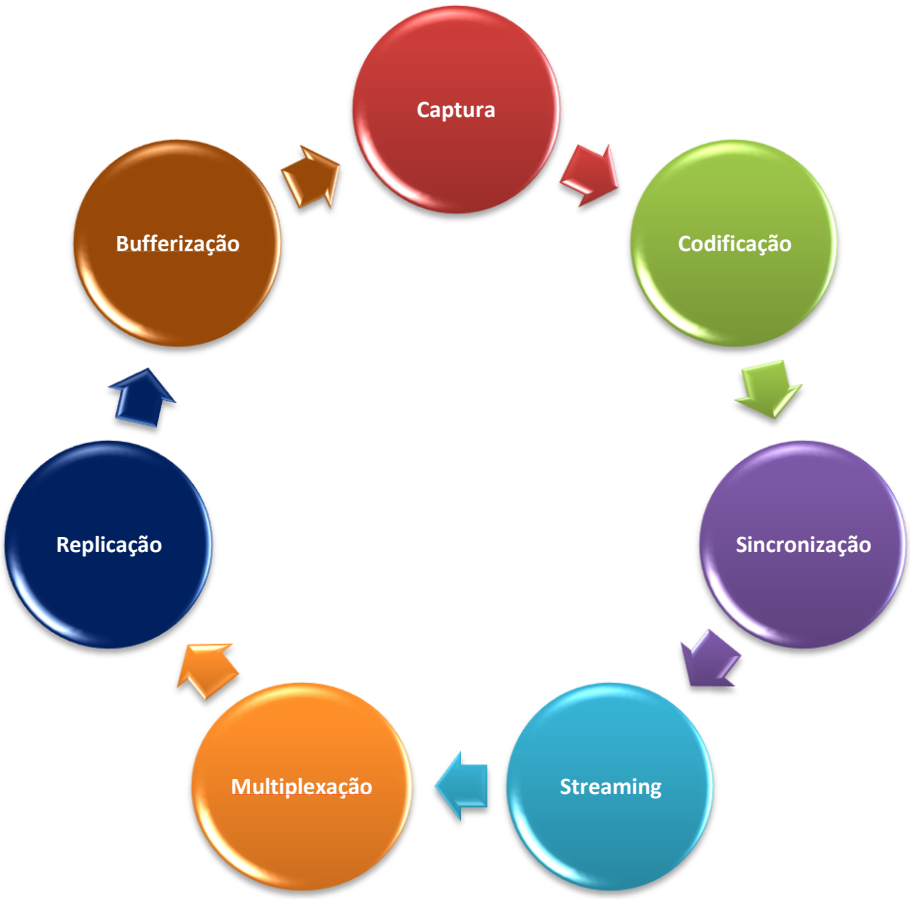
Os sistemas de computação (sistemas de processamento, sistemas operacionais, redes de comunicação etc.) foram desenvolvidos, originalmente, para dar suporte ao processamento e comunicação de dados textuais. Com a evolução tecnológica não só as redes aumentaram em muito seu desempenho, mas também cresceu muito a capacidade de processamento e armazenamento das estações de trabalho. Isso tornou possível o desenvolvimento de sistemas para processamento e comunicação de informações representadas em várias outras mídias além da textual, como áudio e vídeo. O fenômeno da convergência é, em parte, marcado por essa mistura de diferentes tipos de mídia em sistemas integrados de transmissão e processamento de informação (SOARES 2007).

Neste capítulo são apresentados os mais diversos conceitos inerentes a vídeo digital e apresentar algumas ferramentas computacionais que fazem distribuição ou manipulação de conteúdo multimídia. São detalhadas, também, algumas das técnicas de reuso de software, em especial frameworks e o processo de desenvolvimento de software baseado nesta técnica.

2.1 Conceitos Chave

Para o entendimento de aplicações baseadas em vídeo digital é necessário conhecer a natureza dessa categoria de aplicações, seu comportamento e especialmente a dinâmica de sua infraestrutura. Na Figura 3 são destacados os conceitos que julgamos *key concepts* para o entendimento da dinâmica de uma aplicação baseada em vídeo digital nesta seção.

Figura 3: Conceitos chavesobre vídeo digital.



2.1.1 Captura

A captura do conteúdo multimídia consiste basicamente na conversão de um sinal analógico, a exemplo do sinal produzido por uma câmera de vídeo ou de uma guitarra, para um formato digital. O produto dessa conversão é referenciado por *stream* (fluxo) de mídia e pode ser utilizado pela aplicação. Por exemplo, um conjunto de imagens exibidas com o passar do tempo é a maneira mais comum de representar vídeo. Toda representação analógica dessas imagens tem sua analogia digital em um formato específico. A princípio, as informações necessárias para gerar um vídeo são suas dimensões (número de linhas e colunas) e a taxa de quadros. A captura de vídeo digital consiste em transformar imagens analógicas obtidas a partir de um dispositivo como câmeras de vídeo em conteúdo digital para que possam ser utilizadas pela aplicação (MORRIS 2000).

2.1.2 Codificação

Codificação é o conceito de reduzir o volume de dados multimídia sem perder a qualidade da percepção humana da informação. Essa redução pode ser alcançada de duas maneiras: a primeira é por remover a redundância, descartando a informação repetida e a segunda é por descartar a informação perceptivelmente insignificante. A codificação de áudio, vídeo ou imagens é necessária considerando o volume de dados necessário para armazenar uma amostra. Pode-se listar dentre os formatos de codificação largamente conhecidos os seguintes: de áudio o MP3 e o AAC, de vídeo MPEG-2 e o WMV e de imagens JPEG e PNG. Codificar e decodificar dados é um processo computacionalmente intenso. Historicamente, quando processadores tinham capacidades limitadas, hardware adicional de tarefa específica era necessário para codificar e decodificar mídia. Hoje, porém, com o aumento do poder de processamento, softwares especializados podem desempenhar essa função (MORRIS 2000).

2.1.3 Sincronização

Da definição do termo multimídia derivamos o conceito de sincronização entre a informação codificada em vários tipos de mídia. A sincronização cria relações entre objetos independentes como metadados, mídia, processos e fluxos de dados. Exemplos de sincronização em sistemas multimídia são entre vídeo e texto (legendas em filmes) e entre áudio e vídeo ou figuras. Deve-se destacar a diferença entre a sincronização intra-objeto e inter-objeto. A sincronização intra-objeto refere-se ao relacionamento temporal entre unidades de apresentação em um único objeto de mídia, como, por exemplo, a taxa de frames de uma sequência de vídeo. Por sua vez, a sincronização inter-objeto refere-se ao relacionamento temporal entre diferentes objetos de mídia. Especificações de sincronizações devem incluir as subespecificações intra-objetos e inter-objetos e seus respectivos parâmetros de QoS (Quality-of-Service) (STEINMETZ 2010).

2.1.4 Streaming

Streaming é a entrega de conteúdo multimídia diretamente da fonte para um player. É um processo contínuo sem intermediários de armazenamento. Se o conteúdo tiver sido armazenado em um processo de entrega sob demanda, essa entrega deve ter a taxa controlada

para a reprodução em tempo-real (AUSTERBERRY 2005). Streaming de mídia é fortemente baseado em protocolos de transmissão (UDP, RTP) e formatos de arquivos (MPEG-4, WAV).

2.1.5 Multiplexação

Na codificação de áudio e vídeo são utilizados dois agentes de codificação. Após este processo, são gerados dois fluxos independentes com seus respectivos marcos de tempo para a sincronização intra-objeto. A operação que unifica esses dois fluxos e adiciona os marcos temporais para a sincronização inter-objetos é chamada multiplexação. Dados multiplexados podem ser transmitidos pela rede a outros clientes para reprodução. Para tanto é necessário que seja realizada a operação inversa de demultiplexação, ou a separação de diferentes mídias sincronizadas. A multiplexação reduz grandemente a complexidade dos servidores de mídia na distribuição de conteúdo, uma vez que em vez de vários fluxos, é preciso tratar apenas um (LEE 2005).

2.1.6 Replicação

A distribuição do conteúdo multimídia entre componentes do sistema é de importância crítica. Uma abordagem frequentemente utilizada é a replicação. Replicação consiste em realizar, em um cluster de servidores multimídia, clones de porções do conteúdo gerado e distribuí-los entre os componentes ciclicamente ou segundo alguma política pré-estabelecida. Cada requisição de clientes é enviada ao servidor com menor carga que distribui para ele o conteúdo. O custo em termos de tempo para efetuar a replicação cresce diretamente proporcional à quantidade de destinos que o replicador está enviando (DUKES 2004).

2.1.7 Bufferização

Dados multiplexados recuperados de um dispositivo de armazenamento inerentemente introduzem um atraso na entrega das unidades de dados. Para absorver o atraso durante a reprodução é utilizado o conceito de bufferização. Em termos gerais, bufferização é a retirada de uma unidade de dados para a entrega a um cliente de um buffer enquanto outros processos colocam outras unidades geradas neste mesmo buffer. Isso faz com que o transmissor nunca fique sem unidades de dados para envio. Na recepção, o mesmo conceito pode ser aplicado.

Ao passo que unidades de dados são retiradas de um buffer para a reprodução, outras são colocadas assim que são entregues. Algum espaço de buffer de memória também pode ser usado como um cache para o subsistema de armazenamento subjacente. Em aplicações onde os objetos têm distorcido as freqüências de acesso, o servidor pode armazenar em cache os objetos mais frequentemente acessados na memória para um serviço acelerada. O desempenho do cache é determinado pelo tamanho do cache, a granularidade dos objetos em cache e os algoritmos de substituição (ASTHANA 2006).

2.2 Ferramentas computacionais para a distribuição de fluxos de mídia

Atualmente são várias as ferramentas que tem como objetivo principal fazer a manipulação de mídias. A seguir, é feito uma descrição das principais funcionalidades dessas ferramentas.

A ARTHRON (MELO 2010) tem o objetivo de reduzir ou eliminar a necessidade de se ter pessoas especializadas como responsáveis nas transmissões e simplificar todo o ferramental que normalmente se utiliza em transmissões de áudio e vídeo. Isso se deve, basicamente, por ela apresentar uma interface simples para a manipulação e gerenciamento de diferentes fontes e fluxos de mídia simultâneos pré-gravados ou ao vivo e ter envolvidos apenas computadores, câmeras e/ou projetores como ferramentas de trabalho.

A DICE (THOMPSON) é uma ferramenta que processa vídeo em tempo real, através de filtros. Esses filtros aplicados aos vídeos nada mais são do que a manipulação de pixels que combinados matematicamente pela ferramenta criam novas imagens a partir de um fluxo de vídeo em tempo real. Essa ferramenta foi desenvolvida para a plataforma proprietária Macintosh da Apple e limita-se a inserção de filtros em vídeos em tempo real capturados de câmeras localmente acopladas ao computador, não fazendo uso de algum sistema de transmissão da pilha de protocolos TCP/IP.

O Grass Valley 3000 (GRASS VALLEY) é um equipamento que tem por principal finalidade comutar várias fontes de vídeo em tempo real ou pré-gravadas. Essas fontes de entrada podem ser via cabo composto (analógica) ou digital (serial ou paralela). Com esse equipamento pode-se ter um total de 64 entradas de fontes de vídeo distintas e comutá-las em qualquer saída através do acionamento de botões e alavancas. Apesar de ser um equipamento

altamente profissional, ele limita-se a manipulação de vídeos localmente. Outro entrave é o alto custo na obtenção ou aluguel de tal aparato tecnológico.

SuperCollider (SC) (MCCARTNEY) é um ambiente e linguagem de programação para síntese de áudio e composição algorítmica em tempo real. A partir da versão 3 o ambiente é dividido em servidor e cliente que se comunicam via sockets utilizando a Open Sound Control (OSC) 6 (CNMAT). O servidor SC suporta simples plugins feitos na linguagem de programação C, o que torna fácil o desenvolvimento de algoritmos de manipulação de sons eficientes e todo controle externo no servidor ocorre via OSC. No lado servidor o processo de geração de som se dá através de um executável otimizado em linha de comando chamado scsynth, que na maioria dos casos é controlado pelo ambiente do SuperCollider, mas também pode ser independente. Neste ambiente observa-se a inserção e manipulação de sons na rede em tempo real usando o protocolo TCP/IP. Também é destacado o caráter da licença ser GNU Generic Public Lincense (GNU GPL), ou simplesmente software livre, o que a torna um ambiente propício para ser inserido em outros projetos de software livre, que necessitem a manipulação de áudio em tempo real.

O INTERACT (MANGOLD INTERNATIONAL) é um software usado por muitos pesquisadores que necessitam coletar dados em estudos observacionais de ambientes multimídia. Ele permite que o usuário interaja com seu material de áudio/vídeo, analisando-o através do pressionamento de algumas teclas do computador ou usando o mouse. Através dele é possível manipular partes do vídeo, permitindo entrar em detalhes reais da cena, integrar qualquer tipo de dado externo, como informações fisiológicas ou dados do disco, oferecendo uma ampla gama de possibilidades de visualização e análise, tais como estatísticas, análise de confiabilidade, análise sequencial num intervalo de tempo etc..

O SLTV (TV SOFTWARE LIVRE) é uma ferramenta que captura conteúdo de áudio e vídeo e envia estas mídias para servidores Icecast. Ele possui uma interface gráfica que apresenta o vídeo transmitido, permite a aplicação de efeitos dinâmicos e a configuração dos parâmetros da transmissão. No uso da TV Software Livre (TVSL), foi identificada a necessidade de um software que reúna as tarefas de captura de áudio e vídeo, transcodificação de formatos e transmissão de vídeo (stream) para servidores Icecast 2. Há diferentes métodos de transmissão, utilizando e combinando programas que realizam cada uma das tarefas supracitadas. Assim, neste cenário, está se desenvolvendo o SLTV como o programa principal para o setup de transmissão da TVSL, permitindo a captura e codificação

em tempo real de diferentes entradas de vídeo e áudio, de dispositivos externos de captura como câmeras firewire, webcams USB e placas PCR e DVB.

Para fazer uma análise comparativa entre as ferramentas atuais mais importantes para manipulação de mídia, foram analisados alguns parâmetros que poderão ser usados no processo de especificação de qualquer projeto que envolva mídias digitais, tanto para se verificar se eles se ajustam ao contexto que se quer, se eles têm alguma funcionalidade que precisa ser agregada ou se é necessário desenvolver uma nova ferramenta.

- *Suporte TCP/IP*: uma funcionalidade essencial para a transmissão de mídias, visto que ele é o principal protocolo de envio e recebimento de dados. A Arthton, o SC e o SLTV se utilizam dessa funcionalidade para fazer a transmissão das mídias que eles estiverem manipulando.
- *Streaming de Áudio*: em muitos experimentos, muitas vezes, ter o áudio é requisito fundamental para que ele ocorra com sucesso. Dentre as ferramentas atuais apenas o DICE não faz transmissão de áudio, preocupando-se apenas em aplicar filtros ao vídeo.
- *Streaming de Vídeo*: outro requisito que se torna cada vez mais fundamental com o desenvolvimento das redes de computadores é a transmissão de vídeo, tanto em alta definição quanto em baixa. Isso, além de ser um serviço "extra" oferecido, garante uma maior popularidade do experimento. Apenas o SC não faz transmissão de vídeo, preocupando-se apenas com áudio.
- *Gerador de Estatísticas*: adquirir informações da rede é fundamental principalmente quando se quer fazer transmissão de vídeo ao vivo. Nunca se sabe como a rede em determinado lugar estará e ter essas informações é essencial para que se possam fazer os ajustes de codificação do vídeo para evitar a perda excessiva de pacotes. A Arthron e o INTERACT implementam essa funcionalidade.
- *Sistema Distribuído*: trabalhar com vídeo faz com que se seja necessário se ter uma máquina com um poder grande de processamento. Pensando em minimizar isso, algumas dessas ferramentas, como a Arthron, desenvolveram componentes que podem se comunicar de forma remota.
- *Tipo de Licença*: verificar o tipo da licença de um sistema é fundamental para que se possa ou não agregar ou usar suas funcionalidades. Das ferramentas estudas é possível usar livremente e continuar o desenvolvimento da Arthron, do SC e do SLTV.

Tabela 1. Análise Comparativa entre as Ferramentas

	Suporte TCP/IP	Streaming Áudio	Streaming Vídeo	Gerador de Estatísticas	Sistema Distribuído	Tipo de Licença
Arthron	X	X	X	X	X	Livre
DICE			X			Gratuita
Grass Valley		X	X			Proprietário
SuperCollider	X	X			X	Livre
INTERACT		X	X	X		Proprietário
SLTV	X	X	X		X	Livre

2.3 Reuso de Software

José Carlos Cavalcanti (CAVALCANTI 2007) afirma que o reuso de software é o uso de software existente para o desenvolvimento de um novo software. No reuso de software duas decisões estão envolvidas. A primeira é se se deve, ou não, adquirir software para reusar. Sistemas operacionais devem ser comprados, bibliotecas de códigos devem ser desenvolvidas ou compradas, arquiteturas de domínio específico para famílias de produtos devem ser produzidas. Se o software a ser reusado já é possuído como resultado de outra atividade, esta decisão é desnecessária. Uma segunda decisão é se se deve, ou não, reusar software em instâncias particulares. A questão é: o desenvolvedor deve escrever uma rotina, ou deve buscá-la na Internet? Justamente pelo fato de que o processo de reuso de software envolve encontrar software, entender como reusá-lo e, talvez, modificá-lo antes de ser de fato reusado, pode ser mais atrativo para redesenvolver.

Reuso de software traz benefícios que irão permitir índices melhores de produtividade, produtos de melhor qualidade, consistentes e padronizados, redução dos custos e tempo envolvido no desenvolvimento de software, maior flexibilidade na estrutura do software produzido, facilitando sua manutenção e evolução. Permitem, também, o uso efetivo dos especialistas no desenvolvimento de artefatos reutilizáveis, conformidade aos padrões (por exemplo, fazendo com que os usuários cometam menos erros ao utilizarem uma

interface familiar) e desenvolvimento acelerado, ou seja, economia no tempo de desenvolvimento e validação.

2.3.1 Frameworks

O processo de projeto, na maioria das engenharias, baseia-se em sistemas ou componentes já existentes. Da mesma forma, o reuso de software, o processo de criação de sistemas de software a partir de outros existentes e não “do zero” (KRUEGER 1992), tem sido uma das principais metas da engenharia de software por décadas. Abstração desempenha um papel central na reutilização de software. Abstrações concisas e expressivas são artefatos de software essenciais que devem ser efetivamente reutilizadas com o intuito de reduzir o esforço necessário para ir do conceito inicial de um sistema de software até implementações concretas.

Com o advento do paradigma de programação orientação a objeto foi possível a criação de frameworks de aplicação. Os proponentes do desenvolvimento orientado a objeto sugeriram que objetos eram a abstração mais apropriada para reuso. Contudo, a experiência mostrou que estes são muitas vezes especializados demais para uma aplicação específica. Tornou-se claro, entretanto, que abstrações maiores poderiam apoiar melhor o desenvolvimento orientado a objetos. Estes são os frameworks.

Existem diversas definições formais para framework. Ralph Johnson (JOHNSON 1988) define framework como um conjunto de classes que personificam um design abstrato de soluções para uma família de problemas relatados. Analogamente, Erich Gamma (GAMMA 1994) trata o conceito como um conjunto de classes colaborativas que compõem um projeto reutilizável para uma categoria específica de software. Em geral refere-se a um conjunto de código e bibliotecas que fornece funcionalidades comuns a uma inteira classe de aplicações.

Existem diferentes classificações para frameworks na literatura. Segundo Shimidt e Fayad (SCHMIDT 1997) existem três classes de frameworks: (i) frameworks de infraestrutura, (ii) frameworks de integração de middleware e (iii) frameworks de aplicações empresariais.

- Frameworks de infraestrutura de sistemas são os que apoiam o desenvolvimento de infraestruturas de sistemas, tornando-as portáveis e eficientes. Alguns exemplos são sistemas operacionais, frameworks de comunicação e frameworks para interface de usuário e ferramentas de processamento de linguagem. Frameworks de infraestrutura do sistema são utilizados principalmente internamente dentro de uma organização de software e não são vendidos diretamente aos clientes.

- Frameworks de integração de middleware são os que apoiam as comunicações e trocas de informações entre componentes. São desenvolvidos para melhorar a habilidade dos desenvolvedores de software modularizar, reusar e estender a infraestrutura do seu software para poderem trabalhar sem problemas em um ambiente distribuído. Exemplos comuns são frameworks ORB (Object Request Brokers), middlewares orientados a mensagens e banco de dados transacionais.

- Frameworks de aplicações empresariais incorporam conhecimento de domínio e apoiam o desenvolvimento de usuários finais. Esta classificação engloba vários domínios de aplicação, tais como: telecomunicações, aviação, manufatura, engenharia financeira, dentre outros. Frameworks de aplicações empresariais são essenciais para criar softwares de alta qualidade rapidamente, embora sejam caros de desenvolver ou comprar.

Tony Shan (SHAN 2006) por sua vez classifica frameworks da seguinte maneira:

- conceitual: abrange o modelo da arquitetura.
- de aplicação: o esqueleto de uma estrutura para uma aplicação.
- de domínio: projetados para setores específicos de negócio.
- de plataforma: modelos de programação e ambiente de execução.
- de componente: blocos de construção para uma aplicação.
- de serviço: serviços técnicos e modelo de negócio para computação orientada a serviços.
- de desenvolvimento: construção de uma base para criação de uma ferramenta de desenvolvimento.

Aplicações baseadas em uma estrutura de frameworks exigem investimentos iniciais adicionais. Se um framework é comprado, é necessário recursos para a compra e tempo para aprendê-lo. Se for desenvolvido é necessário tempo e recursos, tanto para desenvolvê-lo quanto para ensinar outros a usá-lo. No entanto, a promessa de um aumento significativo na

produtividade e redução no tempo de colocação no mercado faz o investimento valer a pena na maioria dos casos. Além disso, é reduzido o risco de introduzir novos erros, uma vez que a solução já é consagrada.

A arquitetura de software orientada a objetos utiliza como blocos primitivos objetos e classes. A prática divide a granularidade dos sistemas em três níveis: nível de classe, nível de framework e nível de componente. Para pequenos sistemas, objetos e classes são suficientes para descrever a arquitetura de um sistema. No entanto, quando um sistema se torna maior, mais classes estão envolvidas em sua arquitetura, e abstrações de nível mais alto são necessárias na sua implementação. Sistemas de médio porte podem ser descritos por um conjunto de frameworks colaborativos e de extensões de frameworks (RIEHLE 2000).

Frameworks são usualmente utilizados em um largo contexto na arquitetura orientada a objetos. Desenvolvedores de aplicações em certos domínios tem tido sucesso com o uso de frameworks. Os primeiros frameworks orientados a objetos (MacApp e Interviews) surgiram no domínio de interfaces gráficas de usuário. O Microsoft Foundation Classes, um framework também na área de interfaces gráficas tornou-se um framework padrão na indústria em criação de aplicações gráficas para PC. A comunicação entre objetos locais e remotos tem sido também contemplada com a presença de frameworks ORB (Object Request Broker) na área, eliminando grande parte do trabalho tedioso, passível de erros e não-portável de se criar aplicações dessa classe (SHIMIDT 1997). Atualmente, frameworks para desenvolvimento web que provêem controle de sessão, persistência de dados de alto nível e templates para layouts são bastante difundidos e utilizados em diferentes linguagens de programação.

Markiewicz e Lucena (MARKIEWICZ, 2001) comentam que o desenvolvimento de frameworks é composto por três grandes estágios: análise de domínio, projeto do framework e instanciação do framework.

A análise de domínio tenta descobrir os requisitos de um domínio, bem como seus possíveis requisitos futuros. Durante este estágio, os pontos variáveis e fixos do framework são parcialmente definidos.

O projeto do framework define as abstrações do framework e modela seus pontos variáveis e fixos (às vezes utilizando diagramas UML), prevendo a extensibilidade e a flexibilidade proposta na análise de domínio. Neste estágio, padrões de projeto são usados para auxiliar na modelagem.

A instanciação do framework consiste da implementação dos pontos variáveis do framework que são encaixados no mesmo para gerar uma aplicação. Os pontos fixos do framework, por não serem alterados, são comuns a todas as aplicações geradas.

Segundo Fayad, Schmidt e Johnson (FAYAD, SCHMIDT e JOHNSON 1999), o processo de desenvolvimento de um Framework de Aplicação é semelhante a qualquer processo de reuso e visa construí-lo interativamente. Bosch (BOSCH 1999) identifica as seguintes atividades no processo de desenvolvimento de um framework:

- Análise de Domínio (SCHÄFER 1994): tem como objetivo descrever qual domínio o framework abrange. Para capturar os requisitos e identificação de conceitos, pode-se referir a aplicações já desenvolvidas no domínio, especialista do domínio e normas existentes para o domínio. O resultado da atividade é um modelo de análise de domínio, contendo os requisitos do domínio, os conceitos de domínio e as relações entre esses conceitos.

- Projeto Arquitetural: tem o modelo de análise de domínio como entrada. O designer tem que decidir sobre um estilo adequado à arquitetura subjacente ao framework. Com base nesta concepção o design de alto nível do framework é feito. Exemplos de estilos arquitetônicos ou padrões podem ser encontrados nos trabalhos de Shaw e Garlan (SHAW e GARLAN 1996) e Buschmann (BUSCHMANN 1996).

- Design do Framework: o alto nível do framework é refinado e classes adicionais são projetadas. Resultados desta atividade são as funcionalidades de escopo, as interfaces de reuso do framework, regras de projeto com base em decisões de arquitetura que devem ser obedecidas e um documento de história do design descrevendo os problemas de design encontrados e as soluções selecionadas com uma argumentação.

- Implementação do Framework: a preocupação é com a codificação das classes abstratas e concretas do framework.

- Teste: é realizado para determinar se o framework fornece a funcionalidade pretendida, mas também para avaliar a usabilidade do framework. É, no entanto, longe de ser trivial para decidir se uma entidade é utilizável ou não. Johnson e Russo (JOHNSON e RUSSO 1991) concluem que a única maneira de descobrir se algo é reutilizável é reutilizá-lo. Para frameworks, isto se resume ao desenvolvimento de aplicativos que usam o framework.

- Geração de Aplicações Teste: para avaliar a usabilidade do framework, o teste de atividade de geração de aplicações está preocupado com o desenvolvimento de aplicações de teste com base no framework. Dependendo do tipo de aplicação, podem-se testar diferentes aspectos e, assim, decidir se o framework precisa ser redesenhado ou que é suficientemente maduro para o lançamento.

- Documentação: é uma das atividades mais importantes no desenvolvimento de um framework, embora a sua importância não seja sempre reconhecido. Sem uma documentação clara, completa e correta que descreve como usá-lo, um manual do usuário e um documento de concepção que descrevem como o framework funciona, vai ser quase impossível o uso do framework pelos engenheiros de software que não estão envolvidos no projeto.

Capítulo

3

Trabalhos Correlatos

Pela classificação de Shimidt e Fayad (SCHMIDT 1997) frameworks para o desenvolvimento de aplicações baseadas em vídeo digital se enquadram na classificação de frameworks de infraestrutura de sistemas. No intuito de avaliar o estado da arte em desenvolvimento de frameworks para apoiar a construção de aplicações baseadas em vídeo digital, apresentam-se algumas contribuições da literatura neste capítulo.

3.1 Um Ambiente Integrado para Multimídia Interativa em Sistemas Multimídia Distribuídos

O trabalho de Aygun (AYGUN 2001) propõe um framework para sistemas multimídia interativos em ambientes distribuídos que inclua funcionalidades de sincronização de mídia, processamento de vídeo e interface de usuário. Com respeito à sincronização de mídia, o framework dá suporte a interações VCR (Video Cassette Recorder) como *backward* e *skip* por meio de regras de sincronização orientadas a eventos ou geradas a partir de expressões SMIL (Synchronized Multimedia Integration Language).

O suporte ao processamento de vídeo é de grande valia no tipo de sistemas em questão, uma vez que reduz significativamente a quantidade de dados a ser transmitida. Esse trabalho propõe a compressão do sinal por DCT (transformada cosseno discreta). A interface de usuário proposta além dos controles VCR traz um controle de alto nível de conteúdo de apresentações multimídia.

3.2 EvalVid - um framework para transmissão de vídeo e avaliação de qualidade

O EvalVid (KLAUE 2003) consiste em um framework e um conjunto de ferramentas para transmissão de vídeo e avaliação unificada da qualidade de transmissão. EvalVid tem uma estrutura modular, tornando-se possível tanger os requisitos dos usuários tanto de transmissão quanto de codificação. Assim, é aplicável a qualquer tipo de esquema de codificação e pode ser utilizada tanto em experimentos em tempo real quanto em simulações.

O EvalVid é um framework de aplicação open source com código fonte e binários disponíveis para download na sua página da web (EVALVID). As funcionalidades são implementadas em puro ISO-C para máxima portabilidade e desempenho e há versões suportadas pelas plataformas Linux, Windows e Mac OS.

Sob o parâmetro do monitoramento da rede e do sistema, o EvalVid provê um mecanismo de determinação de perda de pacotes e frames. A perda de pacotes é calculada com base em identificadores únicos de pacotes. No contexto da transmissão de vídeo, é interessante não apenas a verificação de perda dos pacotes como também a informação do tipo de dados que um pacote contém. Essas informações em conjunto dão a ideia de quanto o sistema está sendo comprometido pelas perdas. Perdas de frames também são determinadas e tratadas. Quando é percebida a perda de um frame, o decoder exibe novamente o último frame recebido. Além das perdas, o EvalVid também é capaz de identificar e tratar atrasos utilizando de buffers estatisticamente otimizados na estrutura de transmissão.

O framework também dá suporte à codificação de vídeo. O vídeo proveniente da fonte (sem codificação), que é originalmente armazenado no formato YUV, é codificado para um formato apropriado (tal como MPEG-4) pelo componente do framework *VideoEncoder* antes de ser transmitido. O EvalVid suporta geração de arquivos nos formatos MPEG-4, H.262 e H.264 (KLAUE 2007). Ao chegar ao componente receptor, é realizado o processo inverso de decodificação pelo componente *VideoDecoder*. Essa operação reduz a quantidade de dados a se trabalhar.

O componente *VideoSender* do framework é o responsável pelo streaming de mídia. Vídeos no formato MPEG-4 recebem um tratamento adicionando informações de rastreio do sistema e o identificador do frame, mas mantendo a conformidade com o padrão do formato. O vídeo pode ser transmitido na rede via UDP e ao chegar ao decodificador essas

informações são extraídas. Como já mencionado, a interpretação dessas informações tem a utilidade de monitorar o desempenho do sistema.

Apesar de incluir funcionalidades de avaliação da entrega de frames a sincronização não tem suporte nesse framework. As publicações relacionadas ao EvalVid também não fazem menção a nenhum componente que gerencie destinos dos fluxos de mídia ou a funcionalidades que dão suporte à aplicações multimídia.

3.3 ViFramework - um framework para dar suporte a aplicações com conteúdo de vídeo distribuído

Langen e Opdam (LANGEN 2010) propuseram um framework dedicado ao desenvolvimento de aplicações de vídeo distribuídas, o *ViFramework.* Seu objetivo é criar uma camada de abstração de middleware, permitindo que os desenvolvedores se preocupem exclusivamente no projeto de aplicações de vídeo distribuído. Faz isso por facilitar a distribuição de uma aplicação de vídeo através da utilização de cápsulas de software. Uma cápsula é a unidade de distribuição, servindo como um container de componentes. Ela é capaz de comunicar-se com outras cápsulas pela rede e prover funcionalidades para seus componentes internos. Os componentes não precisam saber nada sobre a rede ou a localização de outros componentes. As cápsulas podem ser implantadas em um único ou em múltiplos hosts e são capazes de transmitir fluxos de vídeo e dados associados sincronizados em uma rede.

O *ViFramework* é um framework meramente conceitual mas que impõe determinados requisitos pragmáticos tais como ser open source, ter as funcionalidades implementadas em C++, ser suportado pelas plataformas Linux e Windows e possuir uma documentação completa.

Com respeito a monitoramento do sistema e da rede, o componente *Gerenciador de distribuição* recupera as médias das performances do sistema como carga de CPU e uso de memória periodicamente e armazena-as em um repositório. Essas métricas podem ser utilizadas por programadores para desenvolverem futuras aplicações ou interpretadas para uma melhor redistribuição da arquitetura dos componentes do sistema.

O framework dá a possibilidade de se transmitir dados codificados em diferentes formatos. Juntamente com o fluxo vai pela rede o seu identificador e um indicador de codec

utilizado. Esse indicador de codec é utilizado por quem recebe o fluxo para efetuar a descompressão. Entretanto, não se faz menção de nenhum componente que efetue essa codificação (nem transcodificação) do fluxo a ser enviado.

No que toca a transmissão de vídeo, o componente do framework *Gerenciador de Streams* utiliza uma biblioteca proprietária de streaming de vídeo para manipular streams de vídeo chamada ViStreaming. Essa biblioteca é baseada numa arquitetura cliente-servidor e permite configurar múltiplos streams de vídeo entre eles. O framework dá suporte à compressão dos fluxos para caber na largura de banda disponível e metadados podem ser sincronizados com o fluxo de vídeo. Diferentes protocolos de comunicação são utilizados: usa-se o TCP para dados não sincronizados e comunicação de eventos e RTP para streaming de vídeo em tempo real e comunicação.

O *Synchronizer* é o componente do framework responsável por sincronizar os dados recebidos por um cliente. A sincronização é feita tanto entre um fluxo de vídeo e seus dados associados como entre fluxos de vídeos. A sincronização na maioria dos casos significa uma operação de wait, ou seja, uma operação que bloqueia a execução de determinada mídia enquanto uma condição não é satisfeita. A sincronização é feita sempre antes da bufferização. Portanto, o buffer contém apenas dados sincronizados prontos para serem reproduzidos.

O framework também provê suporte para uma plena gerência da distribuição e aplicação. O *Gerenciador de Cápsulas* é o responsável por receber, instalar, remover, inicializar e parar uma capsula. Ele é também quem configura os endereços das fontes e dos destinos das conexões entre as cápsulas e recupera os parâmetros dos componentes. Esse componente é configurado para ser acessado e controlado remotamente e gerencia todas as cápsulas em uma máquina.

3.4 Análise comparativa

Aplicações baseadas em vídeo digital demandam frequentemente determinadas funcionalidades. Streaming de mídia, por exemplo, é um requisito mandatório em sistemas distribuídos e, a partir dele, surge a necessidade de um controle de destinos dos fluxos de mídia. Codificação e compressão de mídias reduz o processamento necessário para reproduzi-las, o volume de dados a ser transmitido e, consequentemente, a largura de banda da rede necessária. Gerenciamento da qualidade do serviço é de grande importância em sistemas minimizando perdas ou atrasos. Sincronização também é um requisito de alto grau de

importância apresentando ao usuário a informação de maneira satisfatória. A presença de funcionalidades que não estejam unicamente relacionadas a sistemas multimídia, mas que frequentemente dão suporte a esses, tais como chat ou servidor de mapas, devem ser levadas em consideração numa análise comparativa entre os frameworks. Ter o framework a característica de ser código aberto aumenta seu potencial evolutivo. Pode-se, assim, estabelecer uma análise comparativa entre os três trabalhos supracitados com base nesses parâmetros.

Tabela 2. Análise Comparativa entre os Trabalhos Correlatos

	AYGUN 2001	KLAUE 2003	LANGEN 2010
Streaming de mídia	Sim	Sim	Sim
Gerenciamento de fluxos	Não	Não	Sim
Codificação ou compressão	Sim	Sim	Não
Monitoramento de rede	Não	Sim	Sim
Sincronização	Sim	Não	Sim
Funcionalidades de suporte à sistemas multimídia	Não	Não	Não

Capítulo

4

STYLO

O STYLO é um framework que tem o propósito de dar suporte aos programadores que desejam desenvolver aplicações que contemple os mais diversos cenários em vídeo digital. Nesta seção, o framework STYLO é descrito e detalhado através da metodologia de desenvolvimento que foi escolhida, da API, da visão da arquitetura, da visão tecnológica e de seus componentes.

4.1 Metodologia de Desenvolvimento

Existem diversas metodologias para a construção de frameworks. Dependendo do cenário, cada uma tem seus pontos fortes e fracos. Assim, o primeiro passo para se propor um framework, é entender estas metodologias e escolher a mais adequada para o STYLO.

Johnson (JOHNSON 1997) estabelece a necessidade de se entender o domínio de aplicações através da observação de exemplos concretos. Daí, uma característica relevante do framework será mais facilmente obtida a partir de casos concretos. Esta abstração é modelada através de um processo top-down. Características similares de várias aplicações são modeladas como abstrações de classes. Classes concretas de hierarquia mais baixa representam a especificação pertinente de cada aplicação. Esta metodologia proposta por Johnson é composta de estágios de Análise de Domínio, Design, Implementação e Testes. Ela é, então, mais adequada quando aplicações estão disponíveis desde o início do processo de design do framework.

A abordagem de Pree et al. (PREE, FAYAD, JOHNSON e SCHMIDT 1999) começa da construção de um modelo de objetos para uma aplicação específica, que é refinado através de sucessivas interações. Para cada interação, os pontos flexíveis (chamados de pontos quentes) do aplicativo são inicialmente identificados. Estes pontos são documentados através de cartões de ponto de acesso que são concebidos e implementados. Depois disso, o framework é testado para determinar quais os requisitos de domínio foram cumpridos. Caso contrário, aperfeiçoamentos são feitos através de uma nova rodada de interações.

Mattsson et al. (MATTSSON, BOSCH e FAYAD 1999) apresenta um modelo com seis estágios para a construção de frameworks. A primeira é a análise de domínio, na qual pontos dos requisitos e conceitos relevantes para o domínio são identificados. Na segunda etapa o projeto arquitetural do framework é criado. Na terceira é feito um refinamento do projeto realizado e, em seguida, na quarta etapa o projeto é implementado. Na quinta etapa os testes são realizados, avaliando os recursos e usabilidade do framework. Finalmente, na sexta etapa, a documentação do framework é produzida, em que um dos documentos gerados é o manual do usuário, fornecendo assim, aspectos da usabilidade para o uso da solução desenvolvida no framework.

A abordagem de Pree fornece um alto número de refinamentos e não foi considerada como uma solução adequada para o STYLO, pois apresenta uma complexidade de desenvolvimento maior do que o necessário. A proposta de Mattsson não apoia a construção de um framework a partir de aplicações existentes. Por isso, ela foi descartada.

Para o STYLO, a metodologia de Johnson foi escolhida devido a experiência anterior no desenvolvimento da ferramenta Arthron, bem como detalhes de desenvolvimento de outras aplicações desenvolvidas na instituição, como o JDLive e o JDVod. Assim, para desenvolver estas aplicações, tem-se especificadas todas as suas necessidades, identificando características comuns entre elas, que é o ponto principal da metodologia de Johnson. Assim, optou-se por essa metodologia considerando os resultados e experiência obtida durando o desenvolvimento da Arthron. Permite-se, então, dar continuidade ao trabalho já iniciado e, com isso, atenua-se a curva de aprendizado do domínio modelado. Então, a observação das funcionalidades presentes nestas aplicações guiou o processo de identificação dos requisitos para a definição do domínio a ser tratado pelo STYLO.

A etapa de análise aborda o levantamento dos requisitos e conceitos para a criação do framework. Os principais requisitos serão detalhados na subseção a seguir. Para a fase de

implementação foi utilizada a IDE NetBeans para a codificação do projeto, o sistema operacional Linux na sua distribuição Ubuntu, versão 11.04, bem como a linguagem de programação Java na versão 6.22. A etapa de testes e validação foi realizada de três formas: primeiramente os testes unitários, onde cada funcionalidade desenvolvida é testada. Em seguida, com um número considerável de componentes prontos, aplicações são desenvolvidas para a verificação de integração entre eles. Por fim, foi feita uma validação do framework com testes realizados com programadores para avaliação da ferramenta.

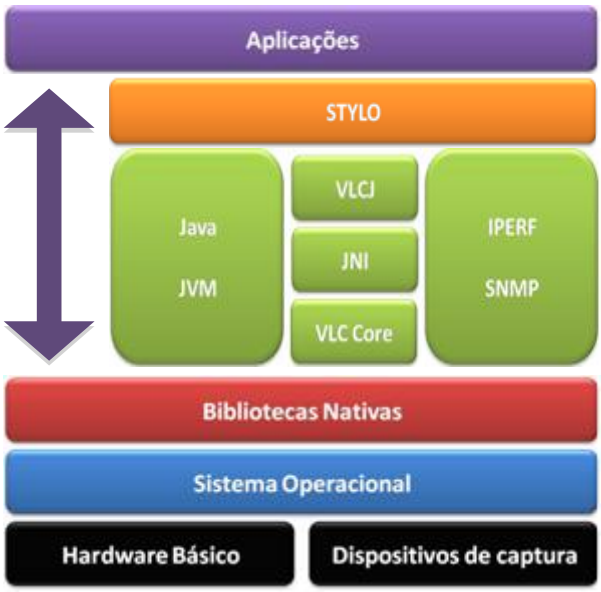
4.2 Visão Tecnológica

O STYLO visa especificar componentes que se comportem de forma genérica e independente da tecnologia que está sendo utilizada. Assim, o programador estará livre para implementar suas classes usando a tecnologia que desejar. Na Figura 4 é ilustrada a visão tecnológica e onde o STYLO se localiza. Uma ponte liga as aplicações às bibliotecas nativas do sistema, ou seja, as aplicações podem acessar funcionalidades diretamente nativas, sem necessariamente passar pelo framework.

Figura 4. Visão Tecnológica do STYLO

Uma instância dessa visão tecnológica é ilustrada abaixo. Com o STYLO é possível usar e customizar a implementação de referência para os componentes que ele especifica. Essa visão é ilustrada na Figura 5.

Figura 5. Visão Tecnológica do STYLO Instanciada



O framework se utiliza de algumas tecnologias como o player VLC, o VLCJ o Iperf e o SNMP, que serão detalhadas a seguir.

VLC Media Player (VIDEOLAN) é um player multimídia altamente portável que suporta diversos formatos de áudio e vídeo (H.264, Ogg, DivX, MKV, TS, MPEG-2, mp3, MPEG-4, aac, dentre outros), de arquivos, mídias físicas (DVDs, VCD, Áudio-CD), placas de captura de TV e muitos protocolos de streaming. Ele também converte arquivos de mídia, transcodifica e funciona como um servidor de streaming utilizando unicast ou multicast e IPv4 ou IPv6. Ele não necessita de nenhum codec ou programa externo para funcionar.

O VLCJ é uma API que faz associações das funções contidas na biblioteca nativa do VLC, escrita originalmente em C, em chamadas de métodos JAVA utilizando para isso o Java Native Interface(JNI).

O Iperf (IPERF) foi desenvolvido pela National Laboratory for Applied Network Research (NLANR) e Distributed Applications Support Team (DAST) como uma moderna alternativa para medir o máximo desempenho de largura de banda via TCP e UDP. Ele gera informações sobre a largura de banda, o atraso médio e sobre quantidade de perdas de pacotes. É de código aberto e roda em diversas plataformas incluindo Linux, Unix e Windows. Assim, utiliza-se essa ferramenta para poder analisar e comparar o desempenho da rede alterando os ambientes de testes e podendo, assim, escolher a melhor configuração possível para execução da performance.

O SMNP (SNMP) é um protocolo de gerência típica de redes TCP/IP, da camada de aplicação, que facilita o intercâmbio de informação entre os dispositivos de rede, como placas e comutadores. O SNMP possibilita aos administradores de rede gerenciar o desempenho da rede, encontrar e resolver seus eventuais problemas, e fornecer informações para o planejamento de sua expansão, dentre outras.

4.3 API

A definição do STYLO é descritaatravés de uma API (Application Program Interface). Nesta API são evidenciadas as funcionalidades necessárias para a implementação de aplicativos baseados em vídeo digital. A Tabela 3 apresenta as funcionalidades que compõem a API do framework STYLO.

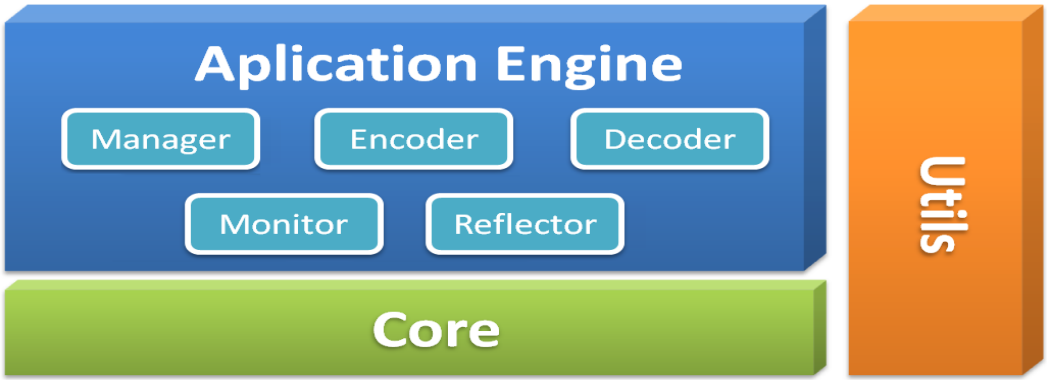
Tabela 3. Funcionalidades Providas pelo STYLO

Funcionalidade	Descrição
Capturar fluxos de mídia	Fazer a captura a partir de câmeras, placas de captura, webcams, stream, arquivo, desktop.
Exibir fluxos de mídia	Exibí-los em diferentes formatos como MPEG, AVI, WMV, FLV, OGG.
Gravar fluxos de mídia	Efetuar a gravação dessa mídia.
Transcodificar fluxos de mídia	Fazer a codificação de um determinado fluxo para outro formato.
Transmitir fluxos de mídia	Fazer transmissão em diferentes protocolos, como UDP, HTTP, RSTP.
Gerenciar fluxos de mídia	Provê suporte para controlar fluxos remotamente, ou seja, pode-se à distância, por exemplo, especificar para onde determinado fluxo vai e em que formato será transmitido.
Replicar fluxos de mídia	Permitir com que um determinado fluxo seja replicado em outros fluxos, podendo-se especificar suas respectivas codificações e destinos.
Sincronizar fluxos de mídias	Permitir fazer a sincronização de fluxos de mídia entre pontos específicos.
Fazer Medição e Monitoramento da rede	Permitir a medição e monitoramento do estado corrente da rede para que se possa verificar qual a codificação ideal do fluxo de mídia naquele momento. Também é possível monitorar os hosts remotos para se certificar de que eles suportam, por exemplo, receber um fluxo em uma taxa de codificação muito alta, se eles têm espaço suficiente para fazer a gravação dessa mídia ou se a quantidade de pacotes que está sendo perdida está fora do esperado.

4.4 Visão Geral da Arquitetura

Para prover as funcionalidades supracitadas, o STYLO organiza sua arquitetura em camadas, como pode ser visto na Figura 6. As principais camadas são: *Application Engine*, *Core* e *Utils*.

Figura 6. Visão Arquitetural do STYLO



A camada *Application Engine* é responsável por prover um conjunto de componentes, métodos, abstrações e funcionalidades que auxiliam o desenvolvedor na construção de sistemas multimídia e, basicamente, possui pacotes com a finalidade de fazer a captura, exibição, codificação, transmissão, gerenciamento de fluxos de mídia e monitoramento da rede.

A camada *Core* refere-se ao controle de agentes externos, como câmeras ou robôs. Assim, com essa camada é possível, por exemplo, adicionar a uma aplicação a funcionalidade de controlar o zoom, a angulação e a direção de uma Câmera IP, bem como controlar os movimentos de um robô, a partir de um protocolo pré-estabelecido.

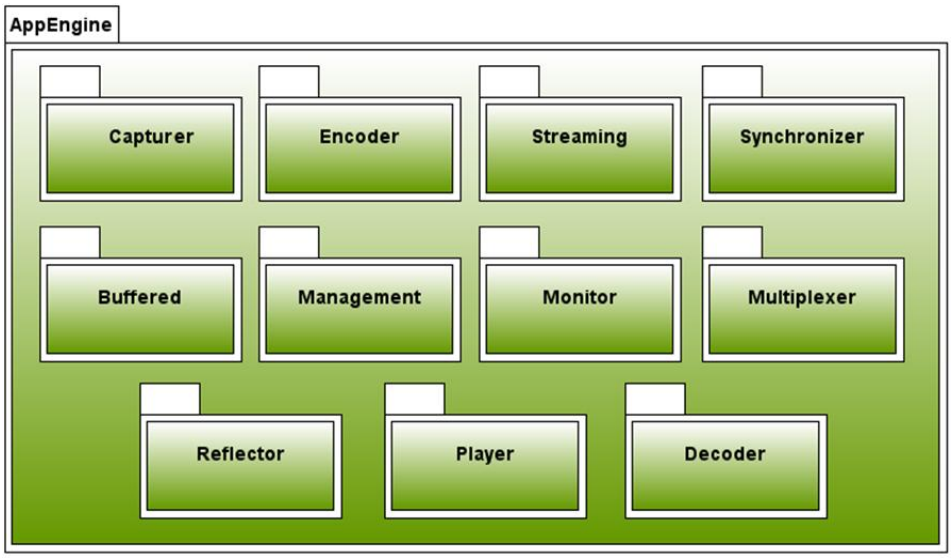
E, finalmente, a camada *Utils* que possui um conjunto de funcionalidades que podem ser adicionadas para incrementar os sistemas baseados em vídeo digital, como segurança e criptografia de mídia, mapa de visualização geográfica, temporizadores, efeitos visuais, registro de atividades, chat, widgets de interface gráfica, dentre outros.

4.5 Visão de Componentes

O pacote AppEngine contém funcionalidades inerentes a sistemas multimídia. Dentro dele pode-se encontrar: o pacote *Captorer*, que dá suporte para a aquisição de fluxos de mídia a partir de dispositivos de captura ou do acesso a arquivos; o pacote *Encoder*, quedá suporte para a codificação/transcodificação de fluxos de mídia; o pacote *Streaming*, quedá suporte para a transmissão de fluxos de mídia em diferentes protocolos, como UDP, HTTO e RSTP; o pacote *Synchronizer*, que dá suporte para que seja feita a sincronização entre fluxos de mídia; o *Buffered*, que dá suporte para armazenamento de dados tanto do lado do consumidor, quanto do produtor; o pacote Management, que provê funcionalidades para que se crie uma aplicação que seja responsável por fazer o gerenciamento e controle de outras aplicações. Assim, por exemplo, é possível manipular os diferentes fluxos enviados e recebidos de forma remota;o pacote *Monitor* é responsável por fazer a medição e o monitoramento da rede. Com isso é possível verificar o máximo de desempenho de largura de banda via TCP e UDP, gerando informações sobre a largura de banda, o atraso médio e quantidade de perdas de pacote. É possível, também, obter informações sobre processamento, memória RAM, quantidade de pacotes enviados e recebidos e espaço em disco; o pacote *Multiplexer/Demultiplexer* dá suporte para que seja feita a multiplexação de fluxos de mídia, ou seja, permitir, por exemplo, a junção de dois fluxos (áudio de um fluxo A + vídeo de um fluxo B) e ter como saída um único fluxo; o pacote *Reflector* dá suporte para que seja possível a partir da recepção de um único fluxo pela rede, replicá-lo para N destinos; o pacote *Player* dá suporte para a implementação das principais funcionalidades de um player, como play, pause, stop, volume, dentre outras; e o pacode *Decoder* provê suporte para a decodificação de fluxos de mídia.

A Figura 7 ilustra a camada AppEngine e os componentes que a formam.

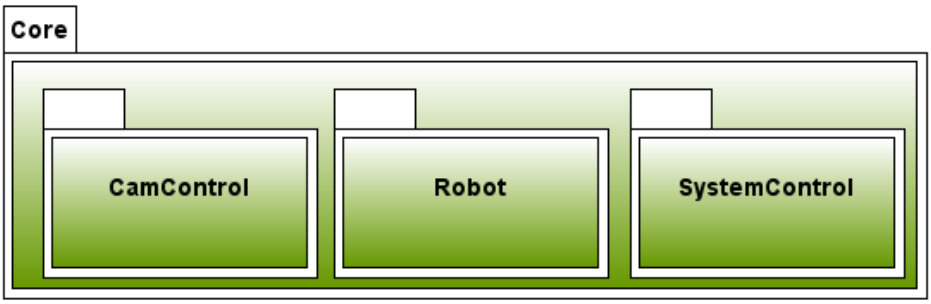
Figura 7. Camada AppEngine e seus Componentes



O pacote *Core* contém funcionalidades para o controle de agentes externos: o pacote *CamControl* fornece abstração de controle de algumas câmeras PTZ e seus principais movimentos, como: pan (direita e esquerda), tilt (cima e baixo) e zoom, de acordo com seus protocolos utilizados; o pacote *Robot* dá suporte para controle de dispositivos, ou grupo de dispositivos, como por exemplo, controlar um robô que tenha uma câmera acoplada. Assim, a partir de um protocolo é possível fazer o seu controle; o pacote *SystemControl* dá suporte para o controle de aplicação via system call, ou seja, dada uma aplicação externa como caixa preta, é possível controlá-la.

A Figura 8 ilustra a camada Core e os componentes que a compõe.

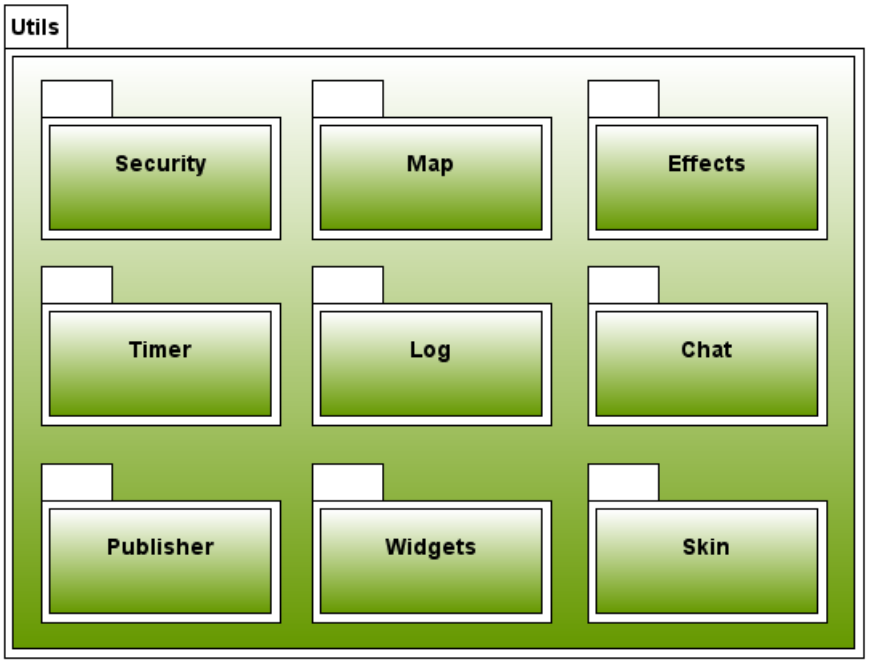
Figura 8. Camada Core e seus Componentes



O pacote *Utils* contém funcionalidades extras que sistemas multimídia podem se utilizar. Este pacote contém o pacote *Security*, responsável por fazer restrições no acesso à aplicação em si ou ao conteúdo, bem como disponibilizar funcionalidades que tem o objetivo de fazer a criptografia do conteúdo transmitido pela aplicação; o pacote *Skin* é responsável por aplicar *look and feels* pré-definidos à aplicação. Desta forma, é possível criar aplicações em domínios diferentes que contemple diferentes exigências de usuário em termos de interface de exibição; o pacote *Chat* é responsável por facilitar a implementação de serviços de comunicação ponto-a-ponto ou multi-ponto entre usuários de um sistema; o pacote *Map* é responsável por disponibilizar um serviço georreferenciado onde é possível localizar um componente de uma aplicação através de um mapa geográfico; pacote *Effects* é responsável por prover facilidades em termos de efeitos visuais na mídia digital. Dessa forma, é possível desenvolver aplicações que disponibilizam efeitos no vídeo, tais como: distorção, rotação, cartoon, sépia, chuvisco, dentre outros; o pacote *Publisher* é responsável por prover a funcionalidade para a criação de páginas web para visualização de vídeos que estiverem publicados na web. Assim, o usuário do STYLO não precisará ter conhecimentos web para configurar ou implementar uma página e disponibilizar para os internautas os vídeos desejados; o pacote *Log* é responsável por registrar o histórico de atividades do usuário efetuadas no sistema; o pacote *Timer* é responsável por permitir ao desenvolver fazer qualquer tipo de programação temporal dentro do sistema, como por exemplo, indicar daqui a quanto tempo um determinado destino irá receber um fluxo de vídeo, ou por quanto tempo uma gravação irá ocorrer; o pacote *Widgets* é responsável por conter componentes de interface gráfica prontos para uso.

A Figura 9 ilustra a camada Utils e os componentes que a compõe.

Figura 9. Camada Utils e seus Componentes

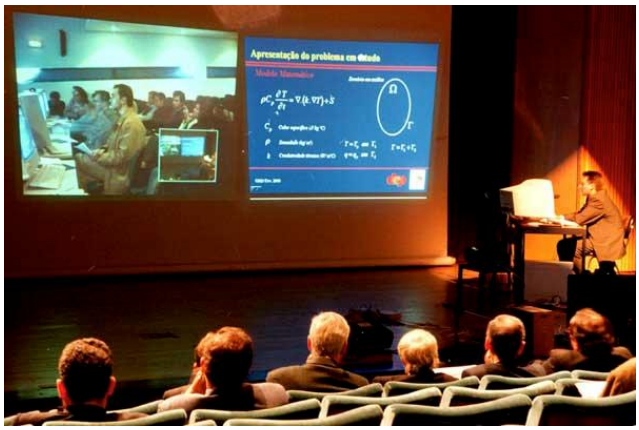


4.6. Cenários de Uso

Como é possível perceber, o uso de aplicações baseadas em vídeo digital pode ser bem abrangente em termos de cenários possíveis. Dessa forma, nesta seção são mostrados alguns cenários com os quais é possível desenvolver aplicações com o uso do framework STYLO. Dentre eles, destacam-se: Educação à distância, Telemedicina e Dança Telemática.

No cenário de educação à distância destacam-se aplicações como tele-educação e transmissão de eventos. Essas aplicações são especialmente interessantes como alternativas educacionais, facilitando a inclusão de novos formatos mais atraentes e motivadores para o aprendiz. Assim, este cenário visa aumentar o acesso ao conhecimento diminuindo barreiras geográficas, flexibilizando o local e o horário das aulas e utilizando diferentes estratégias pedagógicas, atendendo a diferentes perfis e necessidade de desenvolvimento de competências. Nesse cenário destacam-se aplicações voltadas para a transmissão de eventos técnico-científicos ou para tele-aula, com ilustrado na Figura 10.

Figura 10. Cenário de Educação à Distância



O cenário de telemedicina caracteriza-se por ações voltadas para os campos da Telemedicina e Telessaúde. Grandes empresas de tecnologia como Polycom, Tandberg e Cisco estão investindo pesado nestes campos. A Cisco, por exemplo, apresentou na Conferência de 2010 da Healthcare Information and Management Systems Society (HIMSS) a Cisco HealthPresence, uma nova tecnologia de Telemedicina avançada que permite uma ligação remota a médicos e clínicos para consultas médicas, com funcionalidades e tecnologias nunca antes utilizadas. Combinando vídeo em alta definição, excelente qualidade de áudio, e permitindo a transmissão de dados médicos, dando a sensação ao paciente de estar participando de uma consulta presencial.Na Figura 11 é possível ver um médico e seus alunos acompanhando um procedimento cirúrgico em uma sala de telemedicina utilizando um fluxo de vídeo ao vivo como ferramenta didática para incremento da aula.

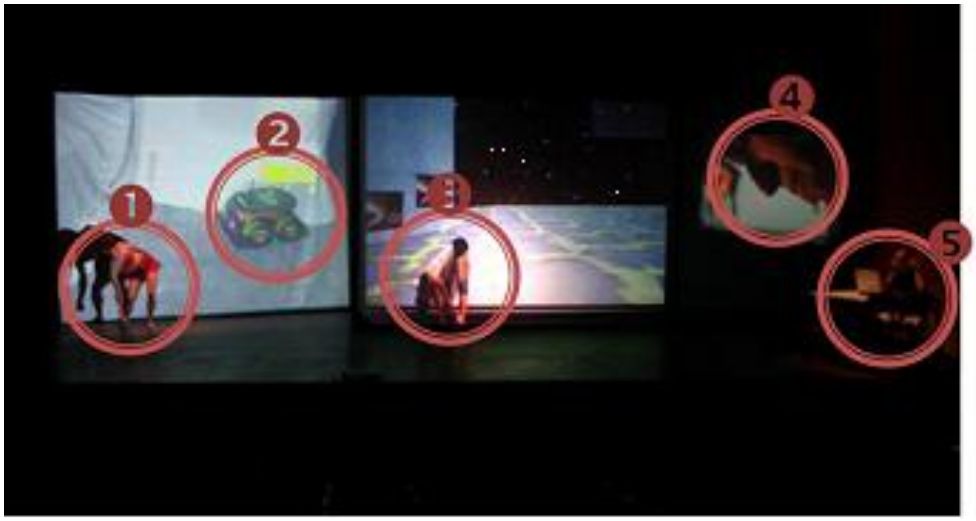
Figura 11. Cenário de Telemedicina

O cenário de dança telemática é ilustrado pela transmissão de eventos artísticos que aproximam e mesclam as áreas de Arte e Tecnologia. Segundo Santana (SANTANA 2003),

as transformações ocorridas em virtude da aproximação das fronteiras entre Arte, Ciência e Tecnologia culminaram em novos formatos de manifestações artísticas. Um exemplo de aplicação nesse cenário é o espetáculo e-Pormundos Afeto (E-PORMUNDOS AFETO).

Na Figura 12 está ilustrada uma visão geral de todos os elementos do espetáculo. Em (1), pode ser visto a dançarina localizado no teatro em Fortaleza, Brasil. A tela central recebe o fluxo de vídeo oriundo de Barcelona, onde é possível ver a dançarina da Espanha (3). Na tela à esquerda, tem-se o fluxo de vídeo de um robô (2) gerado a partir de Natal, Brasil. À direita (4) é possível ver o vídeo transmitido para a Internet. Em (5), pode ser visto o músico, que também estava localizado em Fortaleza.

Figura 12. Cenário de Espetáculo Artístico



Capítulo

5

Resultados Obtidos

Durante o desenvolvimento do STYLO, foi necessário verificar a consistência dos componentes a cada ciclo de desenvolvimento, de tal forma que algumas aplicações resultaram disso. Essas aplicações utilizam funcionalidades de codificação, captura, exibição, transmissão, gerenciamento, e outras funcionalidades que serão detalhadas à medida que elas forem descritas a seguir.

Em uma definição formal, Pressman (PRESSMAN 2006) afirma que:

"Verificação se refere a um conjunto de atividades que garantem que o software implementa corretamente uma função específica e a Validação se refere a um conjunto de atividades diferentes que garante que o software construído corresponde aos requisitos do cliente".

Boehm (BOEHM 1981) diz o seguinte: "Verificação: estamos construindo certo o produto? Validação: estamos construindo o produto certo?". Wallin (WALLIN 2002) afirma que o "propósito da verificação é garantir que o componente de software ou sistemas baseados nele cumpra suas especificações" e que o "propósito da validação é demonstrar que um componente de software ou um sistema customizado atinja o uso desejado em um ambiente desejado".

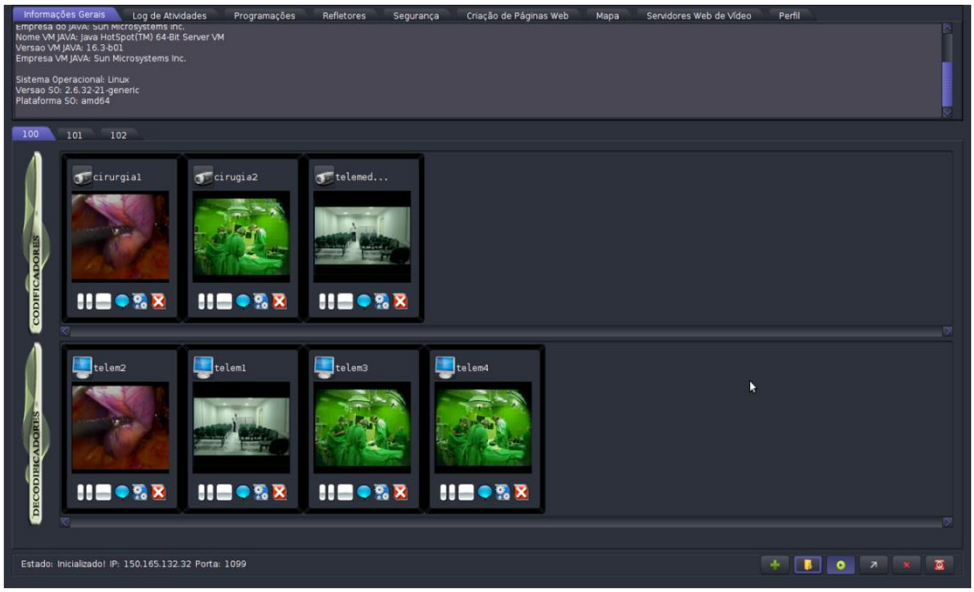
Com isso, neste capítulo é feita a apresentação de algumas ferramentas que foram desenvolvidas como forma de verificar o STYLO e, em seguida, é apresentada a sua validação que foi aplicada com programadores para averiguar a utilização do framework STYLO enquanto artefato de suporte para o desenvolvimento de aplicações.

5.1. Verificação

5.1.1 Articulador

O Articulador (Vide Figura 13) é um componente de software, que compõe a ferramenta Arthron, capaz de fazer o gerenciamento dos fluxos de mídia que ele tem acesso. Assim, com ele é possível controlar para onde (destino) e de que forma (codificação) cada fluxo será destinado, de forma programada ou não. Além, disso também é possível fazer o monitoramento de rede na origem e no destino dos fluxos.

Figura 13. Interface Gráfica do Articulador



5.1.1.1 Detalhes de implementação

Para o desenvolvimento desta aplicação, basicamente, são destacadas as seguintes funcionalidades que foram utilizadas do STYLO:

- Abrir fluxo UDP: no articulador, os fluxos de vídeo que são visualizados através da requisição via UDP dos fluxos que estão sendo transmitidos. Abaixo são ilustrados os códigos antes e depois da utilização do STYLO.
- Programação temporal: provê a abstração para implementação de eventos temporais. É possível, então, implementar um método e especificar dentro de quanto tempo ele será executado. Para isto, cria-se uma classe que herde de *TimerStylo* e implemente o método *action*() (Vide Figura 16). Por fim, quando esta classe for instanciada, passa-

se o tempo que a ação irá levar para ser executada e adiciona-se essa classe ao gerenciador temporal do framework, o *TimerManagementStylo*.

- Look and Feel: é possível aplicar à interface gráfica skins pré-definidos ou customizá-lo de acordo com a necessidade. Assim, tanto para o articulador quanto para o VideoRoom foi aplicado um look and feel azul escuro da seguinte forma:

SkinStylo.getInstance().setSkinStylo(SkinStylo.Themes.DARKBLUE);

SkinStylo.getInstance().setSkinStylo(SkinStylo.Themes.CREAM);

- Mapa Geográfico: no Articulador é possível saber a localização geográfica de cada um dos componentes que estão conectados a ele. Por exemplo, é possível saber onde um determinado VideoRoom (descrito em 5.1.2) está localizado, como ilustrado na Figura 18. Para isso, cria-se um *MapViewerStylo* e adiciona-se a sua localização.

A Fig. 15 apresenta o código da Figura 14 com o uso do STYLO.

Figura 14. Código Legado do Articulador 1.0

```
public void startReceiveStream(int porta) {
    ArrayList<String> args = new ArrayList<String>();

    args.add("-v=2");
    if (!isLinux) {
        args.add("--plugin-path=" + new File("").getAbsolutePath() + File.separator + "plugins")
    }
    args.add("--no-video-title-show");
    args.add("-q");
    args.add("udp://@:" + porta);
    args.add("--marq-opacity="+JVLCSoutGenerator.getLegendOpacity());

    jvlc = new JVLC(args);
    playlist = new Playlist(jvlc);

    String source = "udp://@:" + decoderInformation.getPortToReceiveAudio();
    String sout = ":sout=#duplicate{dst=display,dst=std{access=udp,mux=ts,dst=127.0.0.1:" +porta

    try {
        playlist.play();
    } catch (VLCException ex) {
        ex.printStackTrace();
        Logger.getLogger(DecoderControler.class.getName()).log(Level.SEVERE, null, ex);
    }
}
```

Figura 15. Código gerado com o STYLO substituindo o Código Legado

```
public void startRecieveStream(int port){
    VlcjDecoderStylo decoder = new VlcjDecoderStylo();
    decoder.openUdpStream(port);
    decoder.play();
}
```

Figura 16. Exemplo de Utilização do componente Timer.

```
public class TimeTest extends TimerStylo{

    public TimeTest(long time){
        super(time);
    }

    @Override
    public void action() {
        System.out.println("doSomeThing");
    }

    public static void main(String[] args) {
        TimeTest tt = new TimeTest(10);

        TimerManagementStylo.getIntance().addTimerStylo(tt);
    }

}
```

Figura 17. Exemplos de Resultado da Utilização do Componente Skin.

Figura 18. Utilização do Componente Map.


```
MapViewerStylo map = new MapViewerStylo();
map.addDescriptionWaypoint("VideoRoom", LATITUDE, LONGITUDE);
```

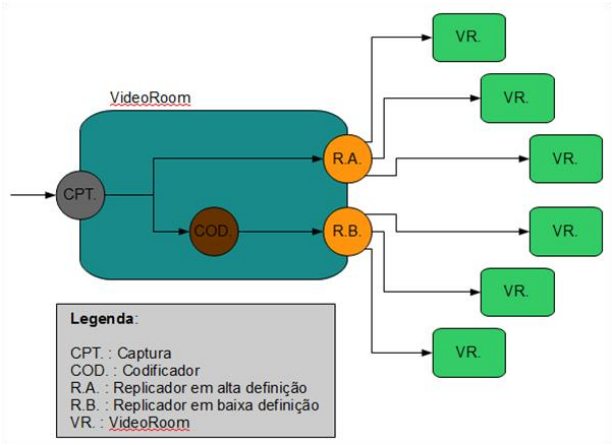
5.1.2 VideoRoom

O VideoRoom é um componente de software capaz de efetuar videoconferências dispensando o uso de MCU (MultipointControl Unit) e compõe a ferramenta Arthron. O componente permite a entrada e saída de fluxos de vídeos e mantém uma relação cliente-servidor com o componente Articulador. Em termos, o maior objetivo do VideoRoom é suprir a necessidade de comunicação multiponto sobre uma infraestrutura de rede comum a baixo custo.

A saída de vídeo consiste em um único fluxo potencialmente capturado a partir de arquivos armazenados; de dispositivos de captura: webcams, placas de captura (pvr) e câmeras IP; da imagem do display do computador; ou de um *stream* UDP de vídeo. Ao mesmo tempo, podem chegar ao VideoRoom múltiplos fluxos de áudio e vídeo provenientes de outros VideoRooms. Dentre esses fluxos que chegam, apenas um deles possui alta definição (acima de 640x480 pixels) e os outros, por questão de limitações na largura de banda da rede, possuem uma qualidade reduzida e são visualizados como *previews*. O usuário pode, entretanto, escolher dentre os vídeos que recebe, qual deles será exibido em alta definição.

Uma vez que mais de um destino remoto é atribuído para um fluxo gerado por um VideoRoom, são utilizados replicadores de conteúdo acoplados ao componente. Esses replicadores recebem um único fluxo e enviam-no para uma lista de destinos dinamicamente alterada. Tendo dito que podem ser transmitidos vídeos de definições alta ou baixa, o processo de transformação de um vídeo com sua qualidade total (a qualidade que é capturado) para outro de qualidade reduzida é realizada por software por um elemento codificador. Nessa etapa as dimensões do vídeo são reduzidas, o formato do arquivo é alterado para um com maior taxa de compressão e reduz-se a taxa de entrega dos dados. Sendo assim, pelo menos dois replicadores são requeridos - um que efetue a distribuição do conteúdo em alta definição e outro que faça o mesmo para conteúdo codificado. A Figura 19 apresenta a estrutura da saída de fluxos de vídeo do VideoRoom.

Figura 19. Visão Esquemática do VideoRoom



A decisão de "o que capturar" é tomada localmente, do lado do VideoRoom. Entretanto, "para quem enviar" e "de quem receber" é determinado remotamente pelo componente Articulador. Um ambiente de videoconferência é configurado quando o Articulador faz ligações entre VideoRooms de modo que eles sejam capazes de se comunicar entre si. A Figura 20 mostra o componente VideoRoom em funcionamento.

Figura 20. Interface Gráfica do VideoRoom

5.1.2.1 Detalhes de implementação

Para o desenvolvimento desta aplicação, basicamente, são destacadas as seguintes funcionalidades que foram utilizadas do STYLO:

- Capturar fluxo de mídia: nesta aplicação isso pode ser feito a partir de um arquivo, câmera ip, stream e placa de captura. Para isso cria-se um *EncoderStylo*, chama-se o método para o que se quer capturar e inicia o processo de captura. Um exemplo é ilustrado a seguir para a captura de arquivo.

Figura 21. Exemplo de Utilização do Componente Encoder

```
EncoderStyloIF encoder = new VlcjEncoderStylo();
encoder.openFile(path);
encoder.play();
```

- Exibir fluxo: para que o vídeo capturado seja exibido, é necessário criar um DecoderStylo e associá-lo ao componente java.awt.Canvas na interface gráfica. Em seguida, especificar a origem da fonte do vídeo e iniciar o processo de exibição.

Figura 22. Exemplo de Utilização do Componente Decoder

```
DecoderStyloIF decoder = new VlcjDecoderStylo(canvas);
decoder.openUdpStream("", 1234);
decoder.play();
```

5.1.3 iReporter

A aplicação de TV consiste em tornar o telespectador de telefornal, mero passivo no modelo de tv atual, em um contribuinte para reportagens, utilizando a Integração de duas redes sociais: Twitter e Youtube. Tendo em visto que a equipe de TV jamais vai conseguir cobrir todos os acontecimentos dos locais onde ela é responsável e o que elas fazem hoje é receber vídeos das pessoas e fazer resumos minúsculos desses vídeos para exibir ao vivo, com esta aplicação será possível o telespectador twittar o vídeo diretamente para a TV e outros telespectadores poderão assistir esse vídeo diretamente da TV e sem cortes. Esses tweets são baseados em *hashtags* para cada reportagem e o usuário basta twittar mencionando o jornal e aplicando a *hashtag* da reportagem escolhida. Na apresentação do jornal os outros telespectadores poderão ver esses vídeos e os comentários feitos pelas pessoas.

A aplicação consiste de uma tela principal, que simula a reportagem sendo exibida (Vide Figura 23). Assim, quando o *"i"* de Interatividade aparecer e o usuário pressioná-lo, a interface gráfica, na qual a reportagem atual está acontecendo, diminui de tamanho e ao lado aparecem os tweets pertencentes a reportagem atual sendo exibida (Vide Figura 24). O usuário pode assistir cada vídeo alí mesmo, mute ou não, ou pode maximizar o tweet.

Figura 23. Interface Gráfica do iReporter sem Interatividade

Figura 24. Interface Gráfica do iReporter com Interatividade

Capítulo

6

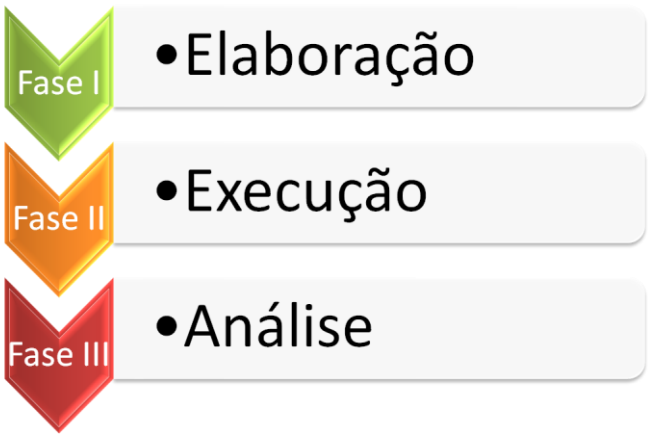
Avaliação Preliminar

Neste capítulo é apresentada uma avaliação preliminar do STYLO enquanto framework de desenvolvimento. No intuito de validar o framework proposto em termos de sua utilização prática utilizamos uma adaptação da metodologia proposta em (SEGUNDO 2011).

6.1. Metodologia Utilizada

A metodologia utilizada é dividida em três fases, como mostra a Figura25, e cada fase possui suas respectivas etapas de execução. A Fase I consiste na elaboração, a Fase II na execução e, por fim, a Fase III é a etapa de análise dos resultados.

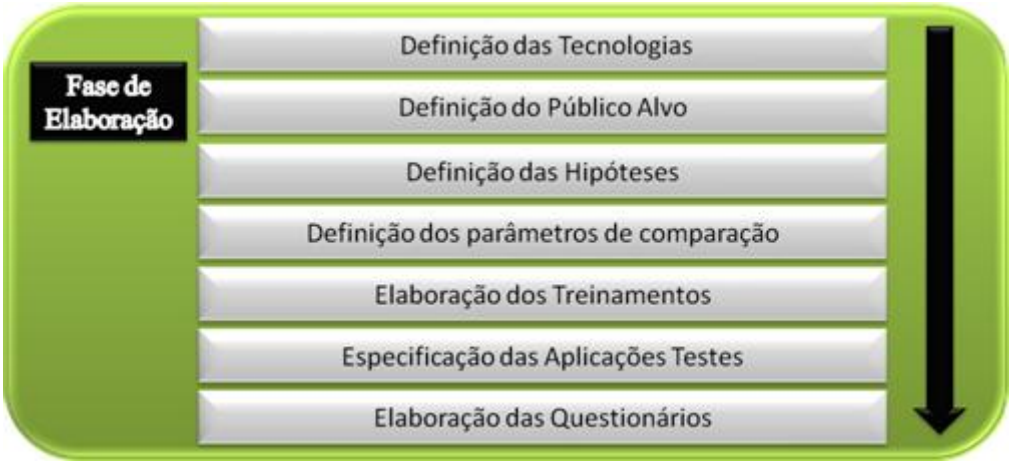
Figura 25 - Fases da metodologia.



Fonte: (SEGUNDO, 2011)

Na fase de elaboração são definidos os aspectos específicos de cada teste, tais como os formulários a serem utilizados na pesquisa e os parâmetros específicos do domínio do framework. A Figura 26 mostra um detalhamento desta fase.

Figura 26 - Etapas da fase de elaboração.



Fonte: (SEGUNDO 2011).

O primeiro passo consiste em definir quais as tecnologias que serão utilizadas nos testes, ou seja, quais outras técnicas serão comparadas ao framework. Devem ser selecionadas tecnologias que proporcionem funcionalidades semelhantes ao framework, para que seja possível implementar aplicações semelhantes com ambas as tecnologias, de modo a compará-las em aspectos quantitativos e qualitativos. Caso não seja feito o desenvolvimento com outras tecnologias, deve-se levar em consideração a experiência que os participantes dos testes têm no escopo da aplicação desenvolvida.

Definir o público alvo para realização dos testes consiste em detalhar a quem se destinam as tecnologias e qual deve ser o perfil dos participantes que irão fazer parte do experimento a ser realizado, de forma que a avaliação deste grupo seja relevante para validar as hipóteses a serem levantadas.

O próximo passo é definir as hipóteses acerca do uso do framework que devem nortear os testes. Elas servirão de base para etapas posteriores, como a definição dos parâmetros a serem utilizados nos testes e as repostas a serem encontradas na análise dos resultados.

Exemplos de hipóteses que podem ser consideradas no contexto de frameworks são:

- O uso do framework facilitou o desenvolvimento das aplicações testes pelos programadores.
- O uso do framework reduziu o número de linhas escritas no desenvolvimento das aplicações testes pelos programadores.
- Os programadores preferiram utilizar o framework a outras tecnologias.

Os parâmetros de comparação devem ser definidos com antecedência.Parâmetros qualitativos serão utilizados para medir a satisfação dos usuários, no caso programadores, em relação ao uso do framework. Desta forma é possível saber se o framework satisfaz os requisitos do programador ou se mudanças devem ser feitas para melhorar o seu uso.

Exemplos de parâmetros que podem ser explorados são:

- Facilidade de uso;
- Facilidade de aprendizado;
- Satisfação com a documentação do framework.

Parâmetros quantitativos também devem ser definidos de forma que a eficácia do framework em relação às outras tecnologias possa ser mensurada. Alguns exemplos são:

- Quantidade de linhas de código escritas;
- Quantidade de Unidades Criadas (classes, aspectos, pacotes, páginas Web, etc.).

Outra etapa é a elaboração de questionários. Os questionários servem para coletar dados dos participantes do teste.

Foi proposta a utilização de um questionário para coleta de informações pessoais que consiste em coletar informações diversas sobre o participante, como nome, escolaridade, idade e outros aspectos pessoais. Informações técnicas também são abordadas para descobrir o grau de conhecimento dos participantes, de forma a averiguar a experiência do programador no domínio do framework, sobre linguagens de programação utilizadas no contexto e experiências dos programadores. O questionário qualitativo abordará os participantes de forma a coletar informações acerca da satisfação e facilidade de uso provida pelo framework.

6.2. Avaliação preliminar do STYLO

Com a metodologia definida e detalhada na seção anterior, foram realizados os testes de validação do STYLO.

6.2.1 Elaboração

Durante os testes, foi utilizada a linguagem de programação JAVA utilizando o Framework STYLO. Utilizou-se a IDE (Integrated Development Environment) NetBeans como plataforma de desenvolvimento e o Ubuntu 11.04 como sistema operacional.

O público alvo do framework são programadores de aplicações multimídia, especialmente as baseadas em vídeo digital e interessados em iniciar o desenvolvimento de programas voltados para esse escopo.

As seguintes hipóteses foram consideradas:

- O uso do framework STYLO facilitou o desenvolvimento de aplicações teste pelos programadores.
- O uso do framework STYLO é intuitivo e supriu as necessidades dos programadores durante o desenvolvimento de aplicações teste.
- Em comparação com experiências anteriores de programadores com outras tecnologias, o framework STYLO reduz o número de linhas de código escritas no desenvolvimento.
- Os programadores que tinham experiência no desenvolvimento de aplicações baseadas em vídeo digital preferem utilizar o framework STYLO, em comparação com outras tecnologias.

Como parâmetros qualitativos foram utilizados:

- Facilidade de aprendizado: avaliar a facilidade dos participantes em aprender a utilizar o framework STYLO.
- Facilidade de uso: avaliar a facilidade dos participantes em utilizar as tecnologias avaliadas.
- Documentação: verificar se a documentação da tecnologia é suficiente para seu uso, ou se melhorias devem ser feitas.

Já os parâmetros quantitativos foram:

- Número de linhas de código escritas no desenvolvimento (Participantes com experiências anteriores devem comparar com funcionalidades semelhantes já implementadas com outras tecnologias).
- Funcionalidades desenvolvidas na aplicação.

O treinamento foi realizado através de aulas expositivas e práticas, com duração total de 8 horas, no qual se levou em consideração as aulas mais o desenvolvimento de aplicações pelos participantes. Foi realizado nos dias trinta e um de outubro e um de novembro de 2011. O primeiro dia foi dedicado para aulas teóricas onde foram abordados os melhores serviços de vídeo da internet, conceitos inerentes a aplicações de vídeo digital, ferramentas computacionais para distribuição de vídeo digital, projetos desenvolvidos no LAViD e uma visão teórica do framework STYLO. O segundo dia foi dedicado para uma apresentação prática do framework e implementação de duas aplicações teste.

O pré-requisito para que os participantes pudessem participar do treinamento foi possuir noção de algoritmos e estruturas de programação, evitando, assim, que pessoas que não tivessem conhecimento em programação pudessem participar do mini-curso. Já alguns conhecimentos eram desejáveis, pois constituiriam participantes mais experientes na área abordada. Tais conhecimentos foram: vídeo digital, linguagem de programação JAVA e orientação a objeto.

A coleta de informações pessoais foi feita através de formulário, criado através da utilização da ferramenta do Google, GoofleForms. A utilização do formulário online facilitou o preenchimento por parte dos participantes e a coleta dos dados por parte dos pesquisadores. O formulário criado pode ser encontrado no Anexo A.

Foi criado outro formulário, encontrado no Anexo B, para análise qualitativa das tecnologias. As perguntas abordadas foram:

1. Você já conhecia a lógica de aplicação desenvolvida? (Domínio de Aplicação)
2. Como você avalia o seu aprendizado da tecnologia utilizada?
3. Como você avalia a prática de uso da tecnologia utilizada no desenvolvimento de aplicações baseadas em vídeo digital?

4. Quais os pontos positivos no uso da tecnologia?
5. Quais os pontos negativos no uso da tecnologia?
6. Comente sua experiência no desenvolvimento.
7. Caso já tenha tido experiência no desenvolvimento de aplicações baseadas em vídeo digital, prefere ou não usar o STYLO?
8. Em comparação com experiências anteriores o STYLO reduz ou aumenta o número de linhas de código para a implementação de funcionalidades semelhantes?
9. O JavaDoc é suficiente como documentação para o uso do STYLO?

6.2.2 Execução

A fase de execução foi iniciada com o recrutamento dos participantes, através de uma chamada pública em listas de discussão da área de computação, como a lista dos alunos do curso de computação da UFPB, a lista dos membros do LAViD, dentre outras.

Os interessados se inscreveram para o mini-curso informando seus dados pessoais, email para contato, o curso e a situação acadêmica atual, o grau de experiência em programação, grau de experiência na linguagem de programação JAVA, conhecimento sobre vídeo digital, dentre outras informações, através do formulário pessoal (Anexo A). Ao fim desta fase, que durou seis dias, vinte e nove pessoas se inscreveram para participar do mini-curso.

Todos os inscritos foram da área de computação (ciência, engenharia, telemática, licenciatura). Apenas dez possuíam conhecimento na linguagem de programação JAVA (considerou-se como "possuir conhecimento" a classificação acima de seis para "grau de experiência com a linguagem" no formulário de inscrição) e apenas oito possuíam domínio ou tinham experiência com desenvolvimento de aplicações baseadas em vídeo digital.

Limitou-se o número de participantes devido ao espaço físico do local do mini-curso e pela quantidade de pesquisadores (dois) que poderiam auxiliar os programadores durante o desenvolvimento das aplicações. Assim, foram selecionados quinze participantes, levando-se em consideração as seguintes prioridades:

- Grau de experiência em desenvolvimento de aplicações baseadas em vídeo digital.

- Grau de experiência na linguagem de programação JAVA.
- Grau de experiência em programação.
- Grau acadêmico.

Assim, foi enviado para os participantes selecionados um email informando local, dia e hora do mini-curso, bem como a especificação da programação. Outro email foi enviado para os participantes que não foram selecionados com o objetivo de agradecer a inscrição e notificar que eles não poderiam participar.

Dos quinze participantes selecionados doze compareceram no primeiro dia do mini-curso. No segundo dia, apenas oito compareceram e participaram da prática. Foi perguntado no formulário de inscrição se seria disponibilizar um computador para o participante. Assim, inicialmente, teríamos que fornecer um computador com uma máquina virtual com o ambiente todo configurado. Porém, durante o mini-curso a máquina virtual não se comportou como o esperado e alguns participantes tiveram que fazer a prática em dupla, visto que a maioria levou o seu laptop.

Utilizou-se a ferramenta Arthron, que foi desenvolvida com o STYLO, para fazer a transmissão online do mini-curso e, assim, poder atingir um número maior de participantes, visto que o espaço do local era reduzido.

6.3. Avaliação preliminar do STYLO – Atividade Prática

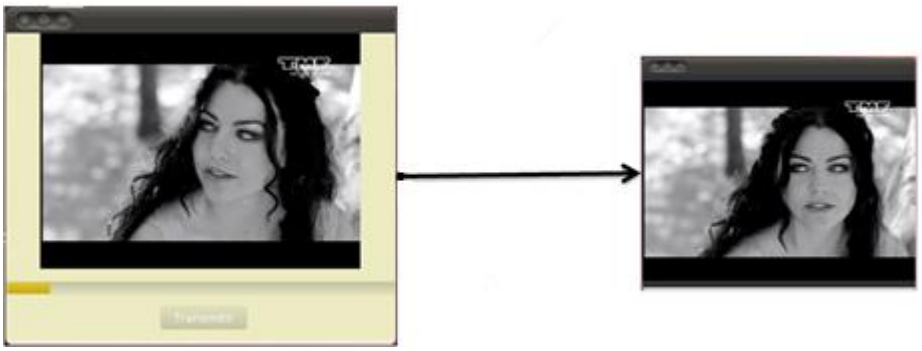
Para a atividade prática, objetivou-se que os participantes implementassem as funcionalidades consideradas principais em aplicações baseadas em vídeo digital: codificação, transmissão e replicação de fluxos.

Assim, inicialmente solicitou-se que eles abrissem um fluxo de mídia, como ilustrado na Figura 27.

Figura 27. Aplicação inicial da prática: abrir fluxo de mídia

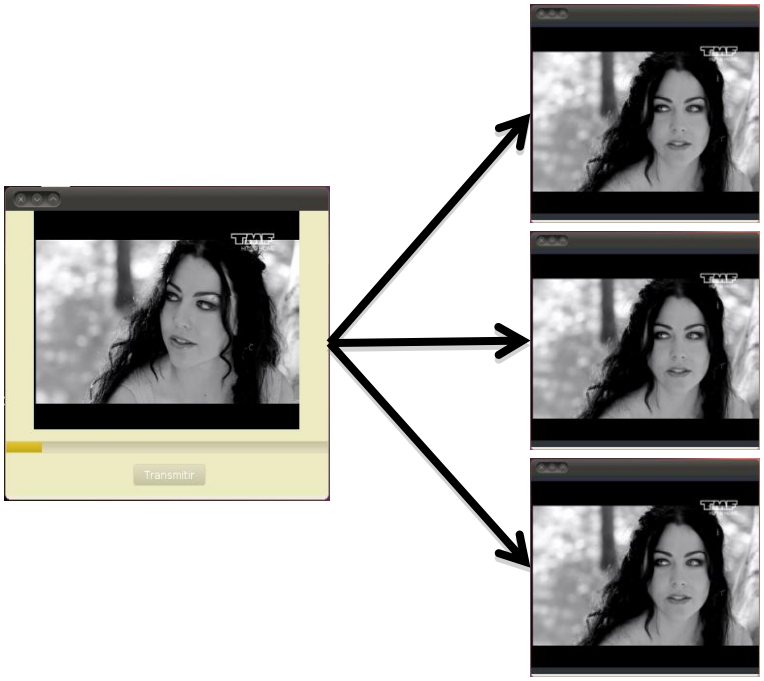
Em seguida, solicitou-se que esse fluxo que estava sendo capturado (de uma webcam ou de um arquivo, por exemplo) fosse transmitido para outra aplicação localizada em um host específico (local ou remoto). Ou seja, um codificador iria fazer a transmissão via UDP, e na outra ponta, uma aplicação iria acessar esse fluxo que a máquina em que ela se encontra estava recebendo. Isso é ilustrado na Figura 28.

Figura 28. Streaming de Fluxo de Mídia



A partir disso, pediu-se que se adicionasse um refletor para que fosse possível fazer uma transmissão para diversos hosts, como ilustrado na Figura 29. Assim, o fluxo que saia do codificador era enviado para o refletor e ele replicaria o fluxo para n hosts (local ou remoto) que iriam receber uma replica desse fluxo.

Figura 29. Reflexão de Fluxos de Mídia



Pediu-se que fosse implementada uma transmissão em http disponibilizando uma URL de acesso para que outras pessoas pudessem acessar aquele fluxo. Assim, o fluxo seria disponibilizado por um servidor e os clientes iriam requisitar esse fluxo. Isso pode ser ilustrado na Figura 30.

Figura 30. Transmissão HTTP do Fluxo de Mídia

6.4. Análise dos Resultados Obtidos

Para a análise dos resultados obtidos observamos as respostas dadas pelos participantes no questionário qualitativo (Anexo B).A partir dessa análise é possível analisar a satisfação dos participantes em relação ao uso do framework STYLO.

Dos doze participantes que foram no primeiro dia, oito participaram do segundo dia, em que se realizou a prática. Desses participantes, metade declarou já ter conhecido a lógica de desenvolvimento da aplicação que foi desenvolvida.

Em relação à documentação da tecnologia avaliada, 80% dos participantes avalioua documentação da API (JavaDoc)como documentação “suficiente” para o aprendizado e desenvolvimento, e os outros 20% como sendo “insuficiente”. A partir de uma documentação considerada “suficiente”, espera-se o aprendizado e o uso do framework tenham uma boa avaliação. Com isso, a Figura 24 ilustra como os participantes avaliaram o aprendizado da tecnologia. Segundo o gráfico de satisfação, o STYLO apresenta nota média de 9,1. A Figura 25 ilustra a avaliação dos participantes em relação à facilidade de uso do framework. De acordo com o gráfico e notas, o STYLO obteve média 8,75. Salienta-se que, dentro dessa amostra, quatro dos oito participantes afirmaram ter experiências com outras tecnologias. A média obtida com apenas esses participante foi de 8,5.

Figura 31.Como você avalia o seu aprendizado da tecnologia utilizada?

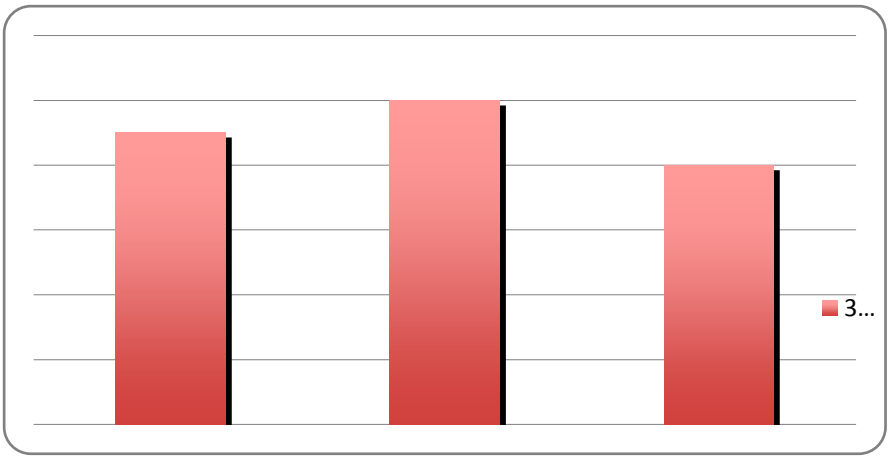
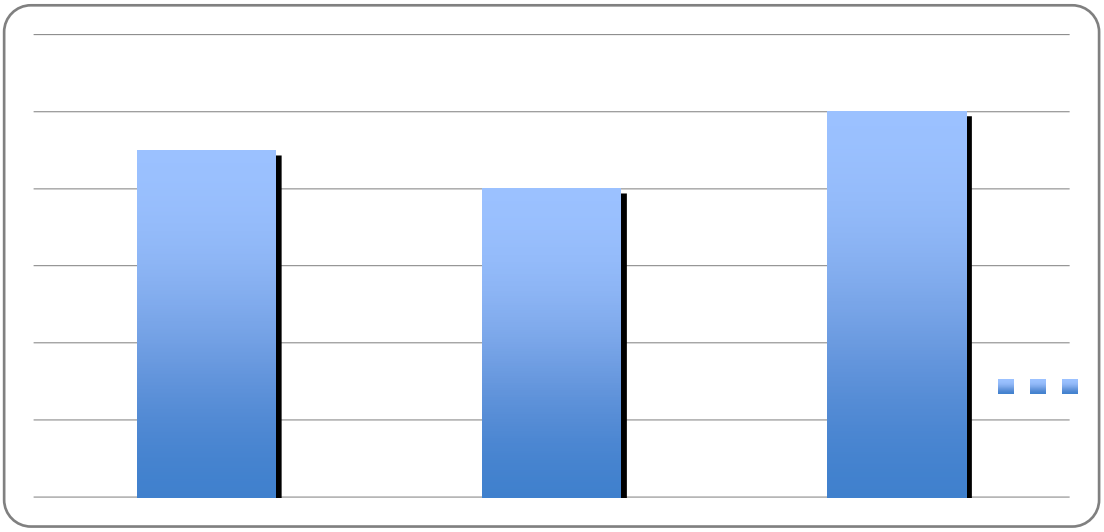


Figura 32. Como você avalia a facilidade de uso da tecnologia utilizada no desenvolvimento de aplicações baseadas em vídeo digital?



Com relação ao número de linhas de código na implementação, perguntou-se aos participantes que, caso já tivessem experiências anteriores com outras tecnologias, avaliassem a implementação de funcionalidades semelhantes. Dos quatro participantes que alegaram já ter experiência, 100% disseram que o STYLO reduz o número de linhas de código.

Com relação à preferência do uso de tecnologias, perguntou-se aos participantes com experiências anteriores qual delas eles prefeririam utilizar. 100% deles disseram que prefeririam utilizar o STYLO perante outras tecnologias que já haviam utilizado. Essa preferência vem principalmente do fato de a grande maioria dos participantes considerarem a documentação do framework suficiente e reduzir o número de linhas de código.

Outra parte do questionário permitia que os participantes falassem de forma aberta sobre os pontos positivos e negativos das tecnologias e suas experiências durante o desenvolvimento.

O principal problema que foi apontado foi a configuração do ambiente para que se pudesse rodar o framework, fazendo-se necessário uma melhor especificação dos requisitos e pendências de configuração. Outro ponto que foi criticado foi a documentação insuficiente ou incompleta, fazendo-se necessário o descrição melhor da API e a criação de aplicações teste para facilitar ainda mais o uso.

Pontos positivos também foram destacados, como a simplicidade, a facilidade de uso e aprendizado e a não necessidade de um alto grau de conhecimento em programação para utilizá-lo.

Quando a experiência no desenvolvimento, o maior número de comentários foi de que o desenvolvimento foi bastante simples. Outros alegaram que não tinham entendido alguns parâmetros e métodos. Isso mostra que é necessário fazer uma refatoração no código para torná-lo mais intuitivo.

De acordo com a análise das respostas analisadas, existem indícios de que as seguintes hipóteses levantadas foram confirmadas. Foram elas:

- O uso do framework STYLO facilitou o desenvolvimento de aplicações teste pelos programadores.
- O uso do framework STYLO é intuitivo e supriu as necessidades dos programadores durante o desenvolvimento de aplicações teste.
- Em comparação com experiências anteriores de programadores com outras tecnologias, o framework STYLO reduz o número de linhas de código escritas no desenvolvimento.
- Os programadores que tinham experiência no desenvolvimento de aplicações baseadas em vídeo digital preferem utilizar o framework STYLO, em comparação com outras tecnologias.

Apesar dos resultados positivos obtidos, novos testes devem ser realizados, com um número maior de participantes, com uma análise estatística mais elaborada e reavaliando alguns parâmetros como o número de linhas de código, que pode variar muito de acordo com o grau de conhecimento do participante na linguagem de programação utilizada.

Capítulo

7

Considerações Finais

O destaque e a popularidade que as aplicações baseadas em vídeo digital estão tendo atualmente tem demandado uma tecnologia de desenvolvimento de software rápida e confiável. Por ser uma área com muitos conceitos e operações recorrentes, independentemente do cenário que se está sendo utilizado, o uso de frameworks no desenvolvimento de tais aplicações reduzem, de fato, o tempo necessário para se desenvolver uma aplicação de software e reduzem o nível de retrabalho dispensado.

A análise comparativa entre os trabalhos correlatos apresentados evidenciou que funções de codificação, sincronização, transmissão, de conteúdo multimídia são indispensáveis em um framework para tais aplicações. Nelas, processamento e rede precisam ser monitorados com respeito a seu desempenho e ter uma forma de gerenciamento dos destinos dos fluxos é imprescindível. Adicionalmente, é desejável que o framework também contemple funcionalidades extras que dão suporte ao desenvolvimento. Uma avaliação do estado da arte na área apresenta o STYLO como uma contribuição importante no campo de frameworks para o desenvolvimento de aplicações baseadas em vídeo digital, uma vez que ele satisfaz as exigências e características desejáveis citadas.

7.1. Resultados Obtidos

Como resultado do trabalho tem-se um framework que dá à comunidade suporte ao desenvolvimento de aplicações que contemple diversos cenários em vídeo digital. Esse framework possibilitara o reuso de código, acelerando o desenvolvimento de tais aplicações,

um bom nível de abstração, suporte para captura, exibição, codificação, transmissão e gerenciamento de fluxos de mídia, suporte para medição e monitoramento da rede. Além disso, facilita a incorporação de funcionalidades que não estão relacionadas propriamente a vídeo digital, porém servem de complemento para aplicações com esse fim, como segurança e criptografia de dados, mapa geográfico, temporizador de operações, customizador de interface gráfica, dentre outros.

Nesta pesquisa, teve-se a oportunidade de desenvolver o STYLO. Disto, lançou-se uma versão inicial de uma API de código aberto, onde os programadores além de se beneficiar das funcionalidades do framework, podem customizá-lo, melhorá-lo e adaptá-lo a situações não previstas. Além da própria API do framework, têm-se como contribuição as aplicações que serviram para fazer sua verificação, em especial a aplicação no contexto de telemedicina.

Durante o WRNP 2011 (WRNP 2011), evento realizado em Campo Grande, MS, foi apresentada pela primeira vez esta ferramenta com aceitação positiva por parte do órgão financiador do projeto e do público visitante. Da mesma maneira, durante o evento que celebrou o aumento da rede acadêmica nacional, a rede Ipê, foi transmitida uma cirurgia realizada no Hospital Lauro Wanderley, da Universidade Federal da Paraíba, acompanhada por estudantes e profissionais de Medicina da Universidade Federal de São Paulo e da Universidade do Tocantins, que puderam interagir com a equipe presente na UFPB.

Como contribuição acadêmica, alguns artigos foram publicados em eventos científicos a partir dessa pesquisa.

- STYLO: Um Framework Voltado para o Desenvolvimento de Aplicações Baseadas em Vídeo Digital. In: Workshop de Teses e Dissertações (WTD)2011, Florianópolis, SC. Simpósio Brasileiro de Sistemas Multimídia e Web (WebMedia).
- Development of a Multi-Stream Tool to Support Transmission in Surgery Applied to Telemedicine. In: ENTERprise Information Systems. Communications in Computer and Information Science, 2011, Volume 221, Part 2, 69-78
- Uma Ferramenta para Vídeo Colaboração em Saúde. In: Congreso Argentino de Informática y Salud, 2011, Córdoba, Argentina. 40 Jornadas Argentinas de Informática.

- A tool for vídeo collaboration in telemedicine. In: IADIS WWW/Internet 2011 (ICWI 2011).
- Tecnologias para o Desenvolvimento de Aplicações Baseadas em Vídeo Digital: Estado da Arte. In: I Workshop de Pós-Graduação (WPG) da I Escola Paraibana de Informática, 2011, João Pessoa.

Em relação aos trabalhos relacionados pesquisados, retomando a Tabela 1 de comparação entre eles, agora com mais uma coluna, de modo a incluir na comparação o framework STYLO, como mostrado na Tabela 4.

Na tabela, percebe-se que o STYLO supre as funcionalidades que alguns dos outros não cobrem, como as funcionalidades de suporte (funcionalidades não relacionadas diretamente a vídeo digital) e cobre funcionalidades consideradas básicas quando se trata de vídeo digital, como streaming e codificação.

Tabela 4. Comparativo entre os trabalhos relacionados e o STYLO

	AYGUN 2001	KLAUE 2003	LANGEN 2010	STYLO
Streaming de mídia	Sim	Sim	Sim	Sim
Gerenciamento de fluxos	Não	Não	Sim	Sim
Codificação ou compressão	Sim	Sim	Não	Sim
Monitoramento de rede	Não	Sim	Sim	Sim
Sincronização	Sim	Não	Sim	Não*
Funcionalidades de suporte à sistemas multimídia	Não	Não	Não	Sim

Por fim, destacam-se os resultados obtidos com os testes com os programadores. Eles apresentam indício que levam a aceitar que o uso do STYLO facilitou e diminuiu o esforço para o desenvolvimento de aplicações baseadas em vídeo digital.

7.2 Perspectivas Futuros

O trabalho realizada nos possibilitou observar na prática o uso de frameworks de implementação como catalisadores da atividades de desenvolvimento no contexto de vídeo digital.

Os resultados obtidos nos motivam a propor melhorias para o próprio framework como adição de funcionalidades ou utilização de padrões de projetos para melhorar a estruturação do código.

Além dessas observações mais restritas ao framework STYLO, este trabalho também pode ser base para a especificação de uma infraestrutura maior de construção de componentes de software, como uma linha de produto de software para aplicações em vídeo digital. Dessa forma, o framework poderia ser expandido para suportar um crescimento voltado a reuso de software e a modelos de componentes.

Referências

(ALEXANDER 1977) ALEXANDER, C., ISHIKAWA, S., SILVERSTEIN, M., JACOBSON, M., FIKSDAHL-KING, I., ANGEL, S. "A Pattern Language". New York, NY (USA): Oxford University Press, 1977.

(APPLETON) APPLETON, B. Patterns and Software: Essential concepts and terminology. Disponível em: <http://www.cmcrossroads.com/bradapp/docs/patterns-intro.html>. Acesso em maio de 2011

(ASTHANA 2006) ASTHANA, A.; KIM, S. H. A longarticle in theEncyclopediaofMultimedia, ISBN: 0-387-24395-X, pp. 580-588, Springer, 2006.

(AUSTERBERRY 2005) AUSTERBERRY, D.The technology of video and audio streaming.2nd ed. Elsevier. Burlington, USA. 2005.

(AYGUN 2001) Aygun, R. Savas. An Integrated Framework for Interactive Multimedia in Distributed Multimedia Systems. ACM Multimedia 2001. September/October 2001. Otawa, Ontario, Canadá.

(BASS 2003) BASS, Len; CLEMENTS, Paul; KAZMAN, Rick. Software Architecture in Practice (2nd Edition). Addison-Wesley Professional, 2 edition, April 2003.

(BOEHM 1981) BOEHM B., Software Engeneering Economics, 1st edition. Prentice-Hall, 1981.

(BOSCH 1999) BOSCH, J.; et al.. Buinling Applicaion Frameworks: Object-Oriented Foundations of Frameworks Design. In: Framework Problems and Experiences. New York: John Wiley & Sons. Cap. 3, p. 55-82.

(BRUNETON, COUPAYE e STEFANI 2002) BRUNETON, E.; COUPAYE, T. ; STEFANI, J.. Recursive and dynamic software composition with sharing. Em: Nuñez, J. H.; Moreira, A. M. D., editors, OBJECT-ORIENTED TECHNOLOGY. ECOOP 2002 WORKSHOP READER : ECOOP 2002 WORKSHOPS AND POSTERS, MALAGA, SPAIN, JUNE 10–14, 2002. PROCEEDINGS, volume 2548 de Lecture Notes in Computer Science, Málaga, Espanha, junho 2002. Association Internationale pour les Technologies Objets, Springer-Verlag Heidelberg.

(BUSCHMANN 1996) BUSCHMANN, F.; et al.. Pattern-Oriented Software Architecture - A System of Patterns, John Wiley & Sons, 1996.

(CAVALCANTI 2007) CAVALCANTI, José Carlos. A Economia do Reuso de Software. http://jccavalcanti.wordpress.com/2007/09/24/a-economia-do-reuso-de-software/

(CLEMENTS 2001) CLEMENTS, Paul C.; NORTHROP, L.. Software Product Lines: Practices and Patterns. SEI Series in Software Engineering. Addison-Wesley, August 2001.

(CNMAT) CENTER FOR NEW MUSIC AND AUDIO TECHNOLOGY (CNMAT). Open Sound Control: Introduction to OSC. Disponível em: <http://opensoundcontrol.org/introduction-osc>. Acesso em: 13 fev. 2010.

(COPLIEN 1996) COPLIEN, J. O. "Software Patterns". New York, NY (USA): SIGS Books, 1996.

(DAVIS 1987) DAVIS, S.M.. Future Perfect. Addison-Wesley, 1987.

(DUKES 2004) DUKES, J.; JONES, J. Using Dinamic Replication to Manage Service Avaibility in a Multimedia Server Cluster in Interactive Multimedia and Next Generation Networks.Springer Link, Grenoble, France. 2004.

(E-PORMUNDOS AFETO) e-PORMUNDOS AFETO. Grupo de Pesquisa Poéticas Tecnolóficas. Disponível em <http://www.poeticatecnologica.ufba.br/site/?page_id=386>. Acessado em 12 de dezembro de 2010.

(FAYAD e JOHNSON 2000) FAYAD, M. E; JOHNSON, R, E. Domain-specific application frameworks: experience by industry. First ed. John Wiley & Sons, 2000.

(FAYAD, SCHMIDT e JOHNSON 1999) FAYAD, M. E.; SCHMIDT, D. C.; JOHNSON, R.E.. Building Application Frameworks: Object-Oriented Foundations of Framework Design. New York: John Wiley & Sons.

(FLATT 1999) FLATT, M.. Programming Languages for Reusable Software Components. PhD thesis, Rice University, Houston, EUA, junho 1999. TR99-345, 158 p.

(GAMMA 1994) GAMMA, E.; HELM R.; JOHNSON, R.; VLISSIDES, J. Design Patterns: Elements of Reusable Object-Oriented Software. Portland: Addison-Wesley, 1994. 395 p.

(GAMMA 1995) GAMMA, E., HELM, R., JOHNSON, R., VLISSIDES, J. "Design Patterns: Elements of Reusable Object-Oriented Software". Reading, MA: Addison Wesley, 1995.

(GIGA VR) RNP (Brasil). GIGA VR – Plataformas para o Desenvolvimento de Aplicações de Realidade Virtual Imersiva e Distribuída sobre Redes de Altíssima Velocidade.

(GRASS VALLEY) GRASS VALLEY (França).GVG 3000 User Manual.Disponível em: www.grassvalley.com/docs/Manuals/DigitalSwitcher/0159_00.PDF>. Acesso em: 22/02/2011.

(GT AVCS) GTAVCS – Grupo de Trabalho em Ambiente de Vídeocolaboração em Saúde. Disponível em: <www.lavid.ufpb.br/gtavcs/> . Acesso em: 18/03/2011

(GT MDA) GTMDA - Grupo de Trabalho de Mídias Digitais e Arte. Disponível em: <www.lavid.ufpb.br/gtmda>. Acesso em: 18/03/2011

(GT TV) RNP (Brasil). GT TV digital. Disponível em: <www.rnp.br/pd/gts2005-2006/tvdigital.html>. Acesso em: 29 jan. 2011.

(GT VD) RNP (Brasil). GT Vídeo digital. Disponível em: <http://www.rnp.br/pd/gts2002-2003/gt- vd.html>. Acesso em: 29 jan. 2011.

(IPERF) IPERF. Disponível em <http://iperf.sourceforge.net/>. Consultado em 23 de Janeiro de 2011.

(JENSEN 1998) JENSEN, J. F. Interactivity: Tracing a new concept in media and communication studies. vol. 19.Nordicom Review. 1998. pp. 185–204.

(JOHNSON 1988) JOHNSON, R. E, FOOTE, B. Designing reusable classes.Journal of Object-Oriented Programming, (S.l.), p.22-35, June/July 1988.

(JOHNSON e RUSSO 1991) JOHNSON, R. E.; RUSSO, V. F.; “Reusing Object-Oriented Design,” Technical Report UIUCDCS 91- 1696, University of Illinois, 1991.

(JOHNSON 1997) JOHNSON, R. E. Frameworks=(components+patterns). Commun ACM 1997;40(10): 39–42. ISSN 0001-0782.

(KINECT) Kinect - XBox.com. http://www.xbox.com/pt-BR/Kinect/GetStarted. Acessoem 9 de julho de 2011.

(KLAUE 2003) KLAUE, J.; RATHKE, B.; WOLISZ, A.; EvalVid - A Framework for Video Transmission and Quality Evaluation, 2003.Proceedings of the 15th International Conference on Modelling Techniques and Tools for Computer Performance.

(KRUEGER 1992) KRUEGER, C. W. Software Reuse. ACM Computing Surveys, Vol. 24, No. 2, June 1992.

(LANGEN, 2010) LANGEN, V., KOEN S. W., OPDAM, M. A framework for supporting distributed video content applications. TechnischeUniversiteit Eindhoven. 2010.

(LEE 2005) LEE, J. Scalable continuous media streaming systems: architecture, design, analysis and implementation. Wiley, 1st edition. New Delhi, Índia. 2005.

(MANGOLD INTERNATIONAL) MANGOLD INTERNATIONAL (Usa). INTERACT. Disponível em: <http://www.mangold-international.com/en/products/interact.html>. Acesso em: 13 fev. 2010.

(MARKIEWICZ 2001) MARKIEWICZ, M. E.; LUCENA, C. J. P.. Object Oriented Framework Development. ACM Crossroads Student Magazine. 2001.

(MATTSSON, BOSCH e FAYAD 1999) Mattsson M, Bosch J, Fayad M. Framework integration problems, causes, solutions. Commun. ACM 1999;42(10):80–7. ISSN 0001-0782.

(MCCARTNEY) MCCARTNEY, James. SuperCollider. Disponível em: <supercollider.sourceforge.net/>. Acesso em: 13 fev. 2010.

(MCINROY 1968) MCINROY, D.. Mass produced software components. Em: Naur, P.; Randell, B., editors, SOFTWARE ENGINEERING, p. 138–155. NATO Science Comitee Report, 1968.

(MELO 2010) MELO, E.; PINTO, A.; SILVA, J.; et al.. Arthron 1.0: Uma ferramenta para transmissão e gerenciamento remoto de fluxos de mídia. 2010.

(MORRIS 2000) MORRIS, T. Multimedia Systems: delivering, generating, and interacting with multimedia. Springer-Verlag London Limited 2000.

(PALHARES 2005) PALHARES, M. M.; SILVA, R. I.; ROSA, R. As novas tecnologias da informação numa sociedade em transição. VI Encontro Nacional de Ciência da Informação. Salvador, Brasil, 2005.

(PESCHANSKI, BRIOT e YONEZAWA 2003) PESCHANSKI, F.; BRIOT, J.-P. ; YONEZAWA, A.. Fine-grained dynamic adaptation of distributed components. Em: Endler, M.; Schmidt, D., editors, MIDDLEWARE 2003: ACM/IFIP/USENIX INTERNATIONAL MIDDLEWARE CONFERENCE, RIO DE JANEIRO, BRAZIL, JUNE 16–20, 2003, PROCEEDINGS, volume 2672 de Lecture Notes in Computer Science, p. 123–142, Rio de Janeiro, Brasil, junho 2003. ACM/IFIP/USENIX, Springer-Verlag Heidelberg. ISBN: 3-540-40317-5.

(PATTERSON 1995) D. A. PATTERSON. Microprocessors in 2020. Scientific American, 273(3):48-51, September 1995. 150th Anniversary Issue.

(PFISTER e SZYPERSKI 1996) PFISTER, C.; SZYPERSKI, C.. Why objects are not enough. Em: Jell, T., editor, Component-Based Software Engineering: 1 St International Component Users Conference Cuc'96, Munich, Germany, Proceedings, p. 165, Munique, Alemanha, julho 1996. Cambridge University Press/SIGS Books.

(POHL 2005) POHL, K.; BÖCKLE, G.; LINDEN, F.. Software Product Line Engineering: Foundations, Principles and Techniques. Springer, 1 edition, September 2005.

(PRESSMAN 2006) PRESSMAN, R. S. Engenharia de Software. 6ª Ed. Rio de Janeiro: McGrw-Hill, 2006.

(PREE, FAYAD, JOHNSON e SCHMIDT 1999) PREE, M.; FAYAD, W., JOHNSON, R., SCHMIDT, D.. Building application frameworks: object-oriented foundations of framework design, 1st. edn. John Wiley & Sons; 1999.

(RIEHLE 2000) DirkRiehle. Framework Design: A Role Modeling Approach. Ph.D. Thesis, No. 13509. Zürich, Switzerland, ETH Zürich, 2000.

(SANTANA 2003) Santana, Ivani. A imagem do corpo através das metáforas (ocultas) na dançatecnologia. Belo Horizonte - Minas Gerais - Brasil. : s.n., 2003. INTERCOM – Sociedade Brasileira de Estudos Interdisciplinares da Comunicação. p. 17.

(SCHÄFER 1994) SCHÄFER, W.; et al. Software Reusability, Ellis-Horwood Ltd., 1994.

(SCHMIDT 1997) SCHMIDT, D., FAYAD, M. Objetc-Oriented Application Frameworks. Communications ofthe ACM. October 1997 / vol. 40, No. 10.

(SEI) Software engineering institute - SEI. Disponível em http://www.sei.cmu.edu/productlines/. Acessado em 27, Jan. de 2011.

(SEGUNDO 2011) SEGUNDO, Ricardo Mendes Costa. Athus: um framework para o desenvolvimento de jogos para TV Digital utilizando Ginga. 2011. Dissertação de Mestrado pelo PPGI (Programa de Pós-Graduação em Informática) da UFPB.

(SHAN 2006) SHAN, Tony, HUA, W.. Taxonomy of Java Web Application Frameworks. ICEBE'2006. pp.378~385. 2006.

(SHAW e GARLAN 1996) SHAW, M.; GARLAN, D.; Software Architecture - Perspectives on an Emerging Discipline, Prentice Hall, 1996.

(SILVEIRA) SILVEIRA, R. M. Redes de alta velocidade e aplicações multimídia. Disponível em: <http://rmav-sp.larc.usp.br/Documentos/RAVeAplic.pdf>. Acesso em: 11 de julho de 2011

(SNMP) NETWORK WORKING GROUP. A Simple Network Management Protocol (SNMP). Disponível em: <http://tools.ietf.org/html/rfc1157>. Acesso em 10 de Fevereiro de 2011.

(SOARES 2007) SOARES, L.F.G., Monografias em Ciência da Computação n° 01/07, Fundamentos de Sistemas Multimídia, Parte 1 - Aquisição, Codificação e Exibição de Dados. Rio de Janeiro. Editor: Prof. Carlos José Pereira de Lucena, janeiro de 2007.

(STEINMETZ 2010) STEINMETZ, R..NAHRSTEDT, K. Multimedia Systems. Springer-verlag New York Inc. 2010.

(SZYPERSKI 2002) SZYPERSKI, C.. Component Software: Beyond Object-Oriented Programming. Component Software. Addison-Wesley Professional, Boston, EUA, segunda edição, novembro 2002. ISBN: 0-201-74572-0.

(THOMPSON) THOMPSON, John Henry. JHT Other work. Disponível em: <http://www.j4u2.com/jht/newwork.html>. Acesso em: 12 fev. 2010.

(TV SOFTWARE LIVRE) Projeto SLTV.Disponível em: <br.gnome.org/TV/ProjetoSLTV>. Acesso em 17 mai. 2010.

(VIDEOLAN) VLC media player. Disponível em <http://www.videolan.org/vlc>. Consultado em 14 de Janeiro de 2011.

(VIN 1994) VIN, H. M. Multimedia System Architecture. Proceedings of the international symposium on photonics for industrial applications.Austin, Texas, USA. 1994.

(WALLIN 2002) WALLIN, C. Verification and Validation of Software Components and Components Based Software Systems – Based Software Systems. Artech House Publishers, 2002.

(WRNP 2011) XXII Workshop da RNP. Disponível em <http://portal.rnp.br/web/wrnp2011/>. Acesso em: 15 de julho de 2011.

Anexo A – Questionário Pessoal

Desenvolvimento de Aplicações Baseadas em Vídeo Digital

Questionário para coleta de dados dos interessados em participar do Mini-Curso: Desenvolvimento de Aplicações Baseadas em Vídeo Digital
* Required

Nome completo: *
Informe seu nome completo.

Email: *
Informe o endereço eletrônico para contato.

Curso: *
Informe o seu curso.

Grau acadêmico: *
Informe a sua situação acadêmica atual.

- ◯ Graduação em andamento;
- ◯ Graduação;
- ◯ Mestrado;
- ◯ Mestrado em andamento
- ◯ Other:

Grau de experiência em programação: *
Informe o seu conhecimento em programação.

	0	1	2	3	4	5	6	7	8	9	10	
Nenhuma	◯	◯	◯	◯	◯	◯	◯	◯	◯	◯	◯	Programador Sênior

Grau de experiência na linguagem de programação JAVA: *
Informe o seu conhecimento em JAVA

	0	1	2	3	4	5	6	7	8	9	10	
Nenhum	○	○	○	○	○	○	○	○	○	○	○	Certificado

Conhecimento sobre vídeo digital:
Informe o que sabe sobre esta mídia.

Conhecimento sobre o desenvolvimento de aplicações baseadas em vídeo digital:
Informe sua experiência.

Necessidade de computador: *
Responder se precisa que a organização disponibilize uma máquina para o participante. Caso possua notebook e o deseje utilizar, responda não.

- ○ Sim
- ○ Não

Submit

Powered by Google Docs

Report Abuse - Terms of Service - Additional Terms

Anexo B – Questionário Qualitativo

Questionário Qualitativo

Questionário para avaliar o grau de Satisfação da tecnologia utilizada no curso.
* Required

Nome: *
Identificar o nome do participante.

Você já conhecia a lógica da aplicação desenvolvida? *
Domínio de Aplicação

- Não, nunca desenvolvi aplicações baseadas em vídeo digital.
- Sim, como testador.
- Sim, como desenvolvedor.

Como você avalia o seu aprendizado da tecnologia utilizada? *
Facilidade de aprendizado

	0	1	2	3	4	5	6	7	8	9	10	
Difícil de aprender	○	○	○	○	○	○	○	○	○	○	○	Fácil de aprender

Como você avalia a facilidade de uso da tecnologia utilizada no desenvolvimento de aplicações baseadas em vídeo digital? *
Facilidade de uso

	0	1	2	3	4	5	6	7	8	9	10	
Difícil de utilizar	○	○	○	○	○	○	○	○	○	○	○	Fácil de utilizar

Pontos positivos: *
Quais os pontos positivos encontrados no uso da tecnologia.

Pontos negativos: *

Quais os pontos negativos encontrados no uso da tecnologia.

Comente sua experiência no desenvolvimento. *

Caso já tenha tido experiência no desenvolvimento de aplicações baseadas em vídeo digital:

Preferência de tecnologia

- Prefere usar o STYLO em relação a outras tecnologias
- Prefere usar outras tecnologias

Em comparação com experiências anteriores:

Número de linhas de código na implementação de funcionalidades semelhantes

- O STYLO reduz o número de linhas de código
- O STYLO aumenta o número de linhas de código

Submit