
**Instituto Federal de Educação, Ciência e Tecnologia da Bahia - IFBa**
Grupo de Sistemas Distribuídos, Otimização, Redes e Tempo-Real - GSORT
Especialização em Computação Distribuída e Ubíqua
INF612 - Aspectos Avançados em Engenharia de Software
Prof.: Sandro Santos Andrade

# ESPECIFICAÇÃO DE TRABALHO PRÁTICO - 2014

## PARTE I – O Trabalho

**Objetivo Geral.** O trabalho prático tem como objetivo a aplicação das técnicas de projeto arquitetural e projeto detalhado centrados em arquiteturas de *software*, aplicados no contexto de um sistema distribuído moderno, com demandas por requisitos tais como: eficiência, facilidade de evolução, gerência de complexidade, escalabilidade e suporte à heterogeneidade. O aluno deverá ser capaz de demonstrar senso crítico nas tomadas das decisões arquiteturais, bem como os benefícios, justificativas e consequências dessas decisões.

**Formato e Produtos a serem Gerados.** O trabalho prático será realizado em dupla e consiste na entrega de um artigo apresentando o projeto arquitetural e detalhado realizados. Os tópicos a serem apresentados no artigo são descritos a seguir, acompanhados de uma descrição do que espera-se que seja encontrado em cada tópico e sugestões para a sua correta realização. O artigo deve seguir o mesmo formato utilizado na disciplina INF615 - Seminários Científicos I (IEEE, templates LaTeX e Word em http://www.ieee.org/conferences_events/conferences/publishing/templates.html).

**Escopo do Projeto.** O projeto deve conter toda a elaboração necessária para que o sistema possa começar a ser desenvolvido. Isto inclui - além da definição da arquitetura e detalhamento de um dos módulos mais importantes – o levantamento das tecnologias de implementação a serem utilizadas (bibliotecas, soluções de *middleware*, *toolkits*, *application frameworks*, etc), a definição do processo de gerenciamento de qualidade a ser adotado e como o sistema será finalmente implantado e executado.

**Objetivos Específicos.** O trabalho deve apresentar propostas de modelos arquiteturais para as *views* obtidas a partir dos seguintes *viewpoints*: *viewpoint* lógico (estrutural ou componente-conector), *viewpoint* de implantação e *viewpoint* de concorrência. Estes modelos devem estar descritos utilizando a notação de modelagem UML. O próximo passo consiste na realização do projeto detalhado de classes para um dos módulos mais importantes do sistema. Tanto as decisões arquiteturais utilizadas nas *views* quanto os *design patterns* aplicados no projeto detalhado devem induzir e favorecer o atendimento das propriedades não-funcionais desejadas. O projeto finaliza com o planejamento da fase de implementação, indicando a possível utilização de *architecture-implementation frameworks*, soluções de *middleware* e componentes e conectores COTS (*Commercial Off-The-Shelf*), bem como soluções para eventuais desafios de implantação da solução.

**Roteiro do Artigo**: a seguir serão descritos os tópicos a serem desenvolvidos no artigo, acompanhados de uma descrição dos resultados esperados.

**1) Introdução**: apresenta uma visão geral do **artigo** (não do sistema): breve descrição do sistema, qual foi o trabalho realizado e os produtos obtidos. O último parágrafo da introdução deve apresentar a estrutura restante do artigo.

**2) *Interactive Digital TV Show***: esta seção apresenta os objetivos do sistema, aspectos importantes do seu ambiente de execução (possíveis integrações com outros sistemas, etc) e os requisitos funcionais e não-funcionais em questão. A seção deve deixar claro quais aspectos funcionais e não-funcionais devem ser corretamente atendidos pela arquitetura projetada.

**3) Projeto Arquitetural**: aqui serão apresentadas as três *views* derivadas para o projeto: estrutural (componente-conector),

de implantação e de concorrência. Para a *view* estrutural os seguintes aspectos devem estar **claramente** descritos: possíveis estilos ou padrões arquiteturais utilizados (isolados ou em conjunto/híbridos), principais componentes com seus respectivos serviços providos e requeridos, principais conectores utilizados (para cada um descrever os conectores básicos utilizados e os valores aplicados nas suas dimensões de variação) e a configuração final obtida. Para cada um destes aspectos é imprescindível explicar qual é o *rationale*.

**4) Projeto Detalhado**: um dos módulos mais importantes, apresentados na *view* estrutural, devem neste momento ter o seu projeto detalhado. Neste projeto, eventuais *design patterns* e princípios de projeto OO aplicados deverão ser apresentados e justificados. Esta seção deve descrever também todas as soluções *COTS* utilizadas (bibliotecas, soluções de *middleware*, *frameworks*, etc). Para aquelas soluções *COTS* que demandarem impactos no projeto detalhado ou projeto arquitetural, tais consequências devem ser relacionadas (indicadas no texto) com as decisões tomadas nestes dois modelos.

**5) Projeto de Implementação e Implantação**: esta seção apresenta o planejamento do mapeamento do modelo arquitetural em artefatos de implementação. Todas as tecnologias facilitadores deste mapeamento, tais como *architecture-implementation frameworks* (MVC, *command-pattern*), soluções de *middleware* (EJB, CORBA, .NET, etc) e componentes e conectores COTS (*Commercial Off-The-Shelf*), devem ser preferencialmente utilizadas e justificadas. A meta não é entrar em detalhes de programação, porém deve-se fornecer indicativos da análise de viabilidade da implementação do modelo arquitetural desenvolvido. Os aspectos complexos do processo de implantação devem também estar presentes nesta seção, justificando as decisões de mapeamento de componentes e conectores em *hosts*.

**6) Discussão e Conclusões**: nesta seção os pontos fortes e fracos devem ser analisados a partir de uma visão comparativa crítica. É preciso **discutir**, ou seja, não é suficiente somente repetir a introdução com outras palavras :)

## PARTE II – O Sistema

**Visão Geral.** A disseminação de conteúdo multimídia em rede e, em particular, em redes de larga escala como a Internet, tem revolucionado a forma como lidamos com tal tipo de informação atualmente. Exemplos atuais, tais como YouTube, rádios *on-line*, TwitCam, composição e execução musical em rede, jogos, dentre outros, são apenas alguns indícios das possibilidades de ampliação da forma como interagimos com conteúdo multimídia nos dias de hoje. Neste contexto, as aplicações de TV Digital têm demandado um grande número de pesquisas, com o objetivo de ampliar o cenário atual de uso deste recurso. O *Interactive Digital TV Show (IDTVS)* - plataforma para disseminação em rede de conteúdo multimídia rico a ser projetada neste trabalho – tem como objetivo disponibilizar um conjunto de serviços de fácil uso para o desenvolvimento de aplicações ricas de TV Digital.

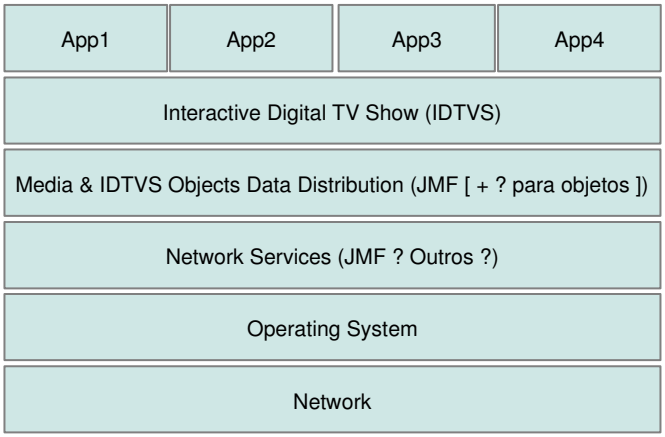


**Figura 1:** possível *technology stack* para o sistema

A figura 1 apresenta um possível *technology stack* para o sistema a ser desenvolvido[1]. O objetivo é proporcionar uma plataforma (biblioteca ? *Middleware* ? *Application Framework* ?) elegante, flexível, heterogênea, escalável, ubíqua e de fácil uso para o desenvolvimento de aplicações modernas para TV Digital. Note que, embora algumas destas propriedades

sejam ambiciosas e potencialmente conflitantes, somente as mais facilmente alcançáveis serão aqui tratadas. Requisitos mais complexos, tais como auto-gerenciamento e alta escalabilidade, serão tratados nas disciplinas do Módulo II da ECDU.

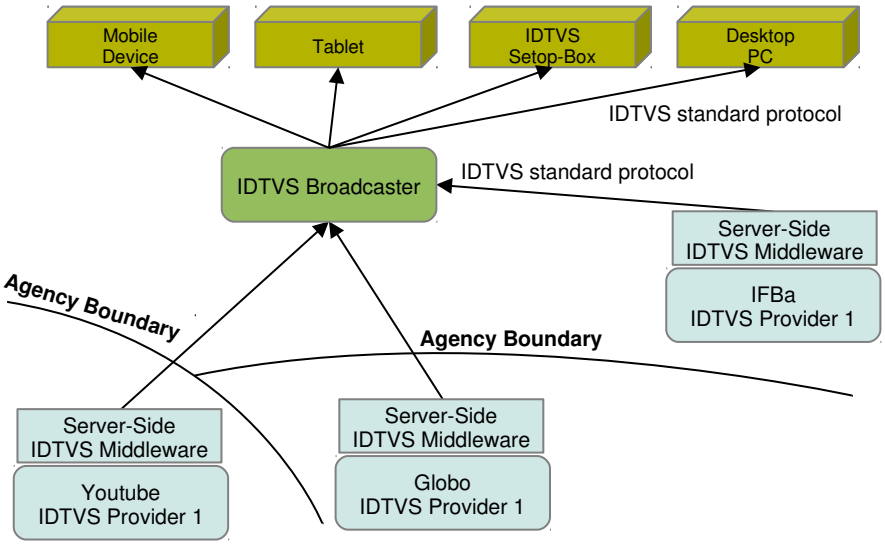


**Figura 2:** possível visão de implantação do sistema

A figura 2 apresenta uma possível visão de implantação do sistema desenvolvido[1]. O aluno deve propor uma arquitetura que contenha todos os módulos necessários para a implementação dos requisitos descritos nos *story cards* abaixo. Dentre estes requisitos, o sistema deve suportar a transmissão de conteúdo multimídia rico sobre a Internet, bem como em ambientes de rede mais síncronos e previsíveis. Aspectos tais como qualidade de serviço e *trustworthness* não serão abordados nas disciplinas do Módulo I.

Vale ressaltar ainda que os alunos estão livres para realizar as decisões arquiteturais e escolha de tecnologias. Note, entretanto, que isto não significa decidir por um estilo arquitetural ou tecnologia com o(a) qual você esteja familiarizado. O objetivo é propor a melhor arquitetura e implementação possíveis para o sistema. O objetivo não é ter um *toy case* e sim uma plataforma que, no futuro, pode se tornar funcional e realmente utilizada.

O sistema prevê a participação de três elementos principais: o *IDTVS Provider*, o *IDTVS Broadcaster* e os *IDTVS Subscribers*. O objetivo de um *IDTVS Provider* é publicar conteúdo IDTVS na rede. Por conteúdo IDTVS entende-se: *streams* de áudio e vídeo, bem como objetos IDTVS genéricos para a ampliação das possibilidades de interação do usuário com o conteúdo multimídia sendo entregue. Desta forma, a Rede Globo, YouTube, IFBa, ou qualquer outra organização, poderiam integrar *IDTVS Providers* na rede e passar a gerar conteúdos multimídia ricos. O *IDTVS Broadcaster* tem como objetivo tratar as questões referentes à obtenção destes conteúdos a partir de *IDTVS Providers* específicos e transmiti-los, de forma eficiente, na rede local onde está instalado. Os *IDTVS Subscribers*, nas suas mais diversas formas (desde computadores *desktop*, até *tablets* e dispositivos móveis), obteriam este conteúdo (possivelmente com a negociação de diferentes qualidades de serviço), a partir do *IDTVS Broadcaster* disponível na sua rede local. Não haverá escolha, por usuário, do conteúdo multimídia a ser exibido por um *IDTVS Provider* específico. O conteúdo sendo exibido pelo *IDTVS Provider,* no momento, é único, proporcionando a utilização de mecanismos mais eficientes de transmissão (por exemplo, *broadcast/multicast*) e permitindo o desenvolvimento de sistemas mais escaláveis.

Um *IDTVS Object* é elemento fundamental para a inclusão de conteúdo rico às *streams* convencionais de áudio e vídeo. Um *IDTVS Object* pode ser qualquer objeto enviado, pelo *IDTVS Provider*, como conteúdo adicional à *stream* em exibição no momento.

1 Não necessariamente é a solução que você vai utilizar no seu projeto. É parte do trabalho do aluno identificar a melhor *technology stack* e diagrama de implantação para o projeto.

**Story Cards.** Seguindo as práticas definidas para o *eXtreme Programming*, os requisitos para o IDTVS são definidos pelos seguintes *story cards*:

**Configuração de conteúdo gravado do IDTVS Provider**

O operador responsável pelo sistema IDTVS que roda no *IDTVS Provider* deseja configurar os conteúdos multimídia sendo exibidos. O sistema deverá exibir a grade de programação do *IDTVS Provider*, indicando quais conteúdos (arquivos) serão executados em dias e horários específicos. O operador poderá carregar um novo arquivo de conteúdo multimídia (previamente gravado), indicando o(s) dia(s) e horário(s) em que será transmitido. O sistema deverá verificar a duração do arquivo informado e verificar se não ocorre choque com o próximo conteúdo agendado para exibição.

O operador poderá carregar diversas versões do mesmo conteúdo multimídia (diferenciadas, por exemplo, pelo *codec* utilizado, *framerate*, profundidade de cor, etc) e indicar as *capabilities* requeridas, do *IDTVS Broadcaster* e *IDTVS Subscriber*, para a execução deste conteúdo. Por exemplo, a *capability* relacionada a poder de processamento, poderá indicar um limiar mínimo (ex: 2.8GHz) necessário para transmissão e execução deste conteúdo pelo *IDTVS Broadcaster* e *IDTVS Subscriber*, respectivamente.

**Configuração de conteúdo ao vivo do IDTVS Provider**

O operador responsável pelo sistema IDTVS que roda no *IDTVS Provider* deseja configurar os conteúdos multimídia sendo exibidos. O sistema deverá exibir a grade de programação do *IDTVS Provider*, indicando quais conteúdos (arquivos) serão executados em dias e horários específicos. O operador poderá indicar que determinado dispositivo de aquisição de vídeo (por exemplo, *web cam*, câmera digital ou placa de digitalização de sinal analógico) deverá ser ligado e seu conteúdo automaticamente transmitido na janela de tempo por ele especificado.

**Configuração dos IDTVS Objects enviados pelo IDTVS Provider**

O operador responsável pelo sistema IDTVS que roda no *IDTVS Provider* deseja configurar os *IDTVS Objects* enviados pelo *IDTVS Provider*. O operador poderá associar *IDTVS Objects* a momentos específicos de cada arquivo multimídia presente na grade de programação do *IDTVS Provider*. O conteúdo destes *IDTVS Objects* não é especificado, permitindo maior flexibilidade nas possibilidades futuras de interação para a plataforma.

**Configuração dos IDTVS Providers retransmitidos pelo IDTVS Broadcaster**

O operador responsável pelo sistema IDTVS que roda no *IDTVS Broadcaster* deseja incluir ou excluir um *IDTVS Provider*. No momento de uma inclusão, o sistema deverá verificar se é possível realizar uma conexão com o *IDTVS Provider* informado e se o *IDTVS Broadcaster* possui todas as capabilities requeridas pelo *IDTVS Provider*.

**Monitoramento do *status* do IDTVS Broadcaster**

O sistema IDTVS que roda no *IDTVS Broadcaster* deve disponibilizar um ambiente de monitoramento da sua execução. Tal ambiente deverá exibir os *IDTVS Providers* sendo retransmitidos no momento, os *IDTVS Subscribers* conectados no momento e uma visão geral do consumo de recursos da máquina (CPU, memória, rede, etc).

| **Descoberta e conexão do IDTVS Broadcaster por um IDTVS Subscribers** |
|---|
| *IDTVS Subscribers* devem ser capazes de descobrir a presença de *IDTVS Broadcasters* na rede local, obter uma lista dos *IDTVS Providers* sendo retransmitidos e decidir se conectar como um *subscriber* do *IDTVS Provider* escolhido. No momento da solicitação da conexão, as *capabilities* disponibilizadas pelo cliente devem ser compatíveis com as *capabilities* requeridas pelo *IDTVS Broadcaster* (obtidas, por sua vez, do *IDTVS Provider*) para a execução do conteúdo multimídia sendo atualmente exibido. A conexão é realizada com sucesso e o *subscriber* começa a visualizar o conteúdo ou uma mensagem informa que as *capabilities* atuais não são suficientes. A solicitação poderá ser realizada em um momento futuro – quando o *IDTVS Provider* estiver enviando um conteúdo com requisitos mais fracos – ou através de um *IDTVS Subscriber* que possua melhores *capabilities*. |

Os *story cards* acima devem ser detalhados em *task cards* e utilizados como base para o projeto arquitetural e detalhado do sistema.

## PARTE III – Exemplo de Utilização das Visões

**Visão Estrutural (Componente-Conector).** A figura 3 apresenta um exemplo da visão componente-conector para um sistema distribuído para troca de mensagens (*chat*). Três *story cards* são estão contemplados neste diagrama: i) consulta, pelo cliente, das salas de bate-papo disponíveis; ii) conexão, envio e recebimento de mensagens pelo cliente em uma sala específica de bate-papo; e iii) configuração, pelo administrador do *chat*, das salas de bate-papo disponíveis, capacidade de cada uma delas, etc.

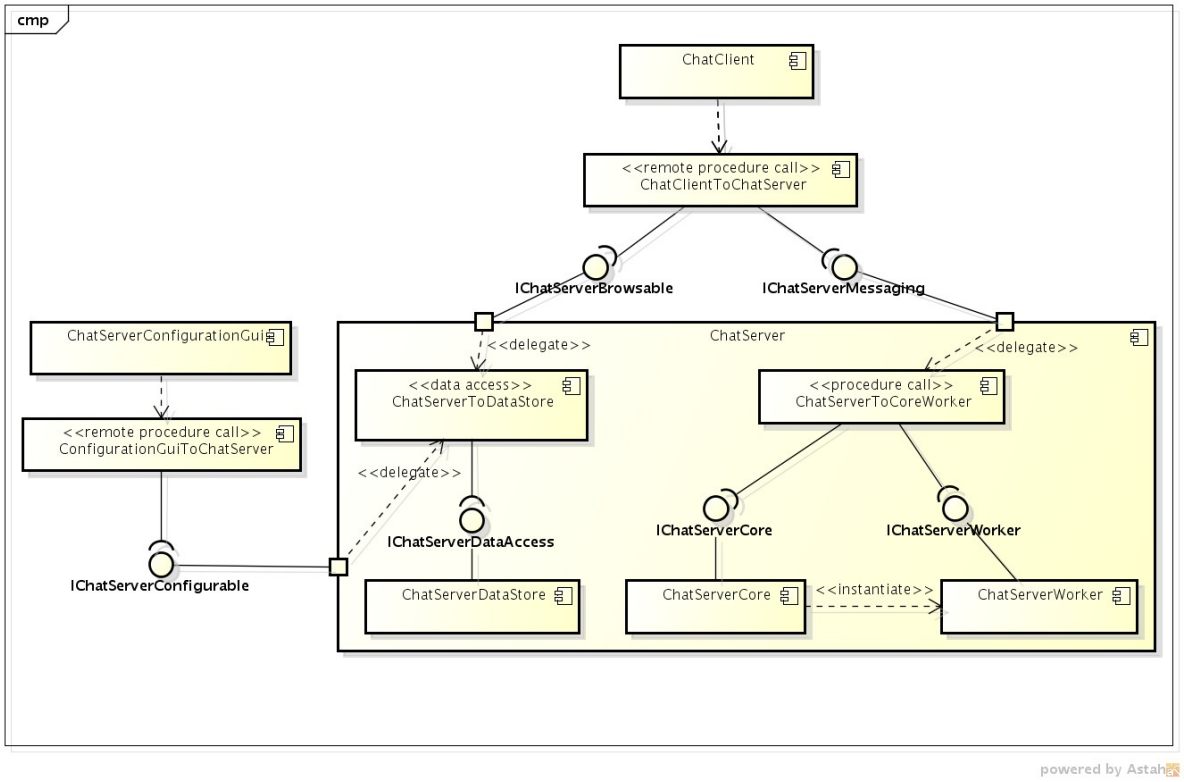


**Figura 3:** exemplo de modelo estrutural

Consulta, pelo cliente, das salas de bate-papo disponíveis. Conforme ilustrado na figura o cliente (*ChatClient*) utiliza um conector do tipo *remote procedure call* para enviar uma requisição de execução do procedimento *getAvailableRooms()* à porta do *ChatServer* responsável pelas operações de consulta. Esta porta, por sua vez, implementa esta requisição através de delegações a um conector do tipo *data access*, que consulta o componente *ChatServerDataStore* através do procedimento *getParameter()* e retorna as informações solicitadas pelo cliente.

Conexão, envio e recebimento de mensagens pelo cliente em uma sala específica de bate-papo. Após conhecer as salas disponíveis, o cliente realiza a solicitação de conexão a uma sala específica. Para isso o conector do tipo *remote procedure call* se comunica desta vez com a porta do *ChatServer* responsável pelas operações de *messaging*. Esta porta, por sua vez, implementa a requisição de conexão através da delegação desta operação ao componente *ChatServerCore*, que cria uma nova instância de *ChatServerWorker* – componente responsável para comunicação com um cliente específico do *chat*. Ao enviar e receber mensagens, o cliente envia requisições de execuções dos procedimentos *sendMessage()* e *receiveMessage()*, implementados pelo *ChatServerWorker*.

Configuração, pelo administrador do *chat*, das salas de bate-papo disponíveis, capacidade de cada uma delas, etc. O administrador do *chat* utiliza uma interface gráfica de usuário para configurar as salas disponíveis, capacidade de cada uma delas etc. Para isso, a interface gráfica de usuário utiliza um conector do tipo *remote procedure call* para enviar solicitações de execução de operações *setParameter()* à porta do *ChatServer* responsável pelas operações de administração. Esta porta realiza a delegação destas operações ao conector do tipo *data access* que acessa o componente *ChatServerDataStore*.

**Visão de Implantação.** Define as características importantes do ambiente operacional de implantação do sistema. Esta visão inclui detalhes sobre os nós de processamento que o sistema requer para a sua instalação (por exemplo, os requisitos mínimos de memória e processamento), as dependências de *software* em cada nó (tais como bibliotecas e plataformas de *middleware* necessárias) e detalhes sobre a rede subjacente. A figura 6 apresenta o modelo de implantação para o sistema do *chat*.

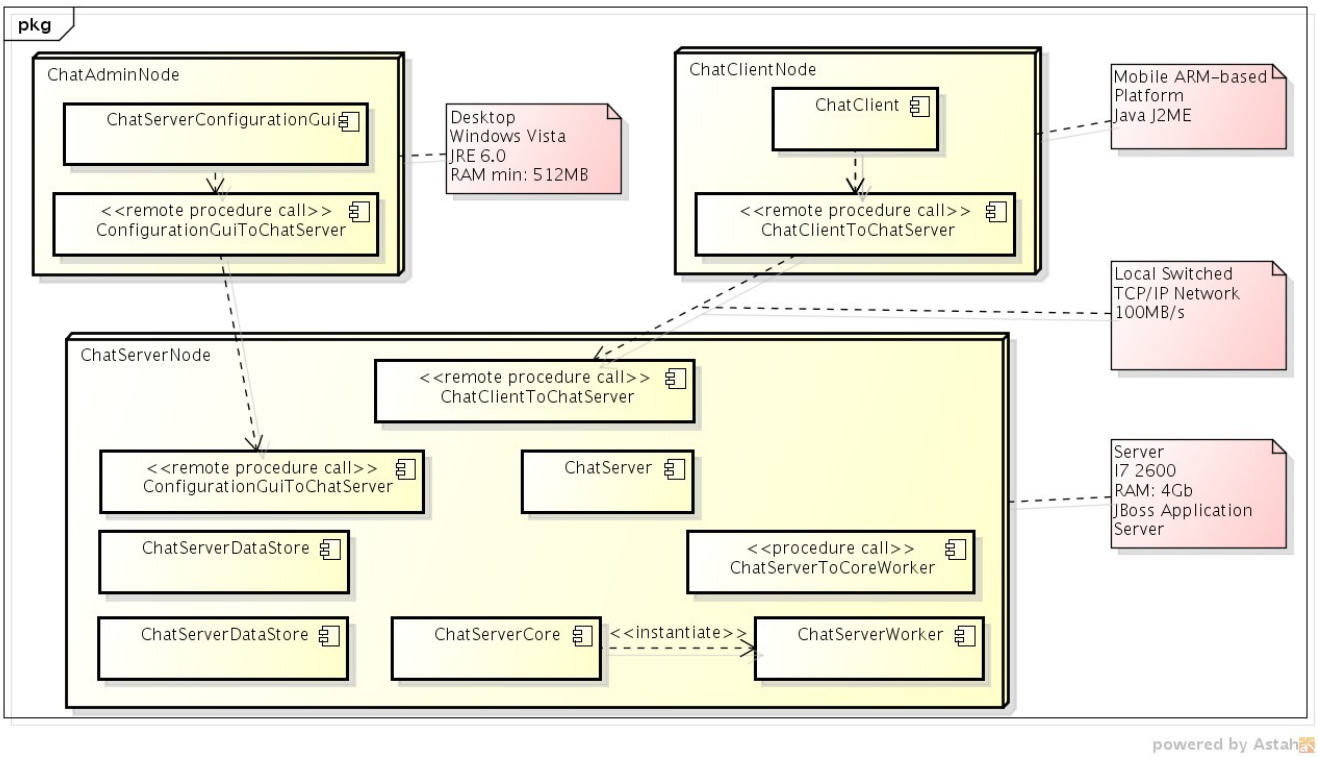


**Figura 6:** exemplo de modelo de implantação

Note que o modelo de implantação deve estar consistente com o modelo estrutural: todos os componentes e conectores do

diagrama estrutural devem estar presentes no diagrama de implantação e o modelo de implantação não deve conter componentes ou conectores não presentes no diagrama estrutural. Perceba também que os conectores que cruzam fronteiras de endereçamento são frequentemente implantados em duas partes (uma *client-side* e outra *server-side*). Isto vale para conectores que realizam interação entre componentes em máquinas diferentes, mas também para aqueles que viabilizam a comunicação entre componentes que residem em processos diferentes de uma mesma máquina. Note também que nesta visão não há necessidade de incluir informações sobre interfaces entre componentes (visto que estas estão presentes no modelo estrutural), somente entre nós de implantação.

**Visão de Concorrência.** Define o conteúdo de elementos de *runtime* do sistema (por exemplo, processos do sistema operacional) que empacotarão os elementos funcionais do sistema. A figura 4 apresenta o modelo de concorrência para o sistema de *chat*.

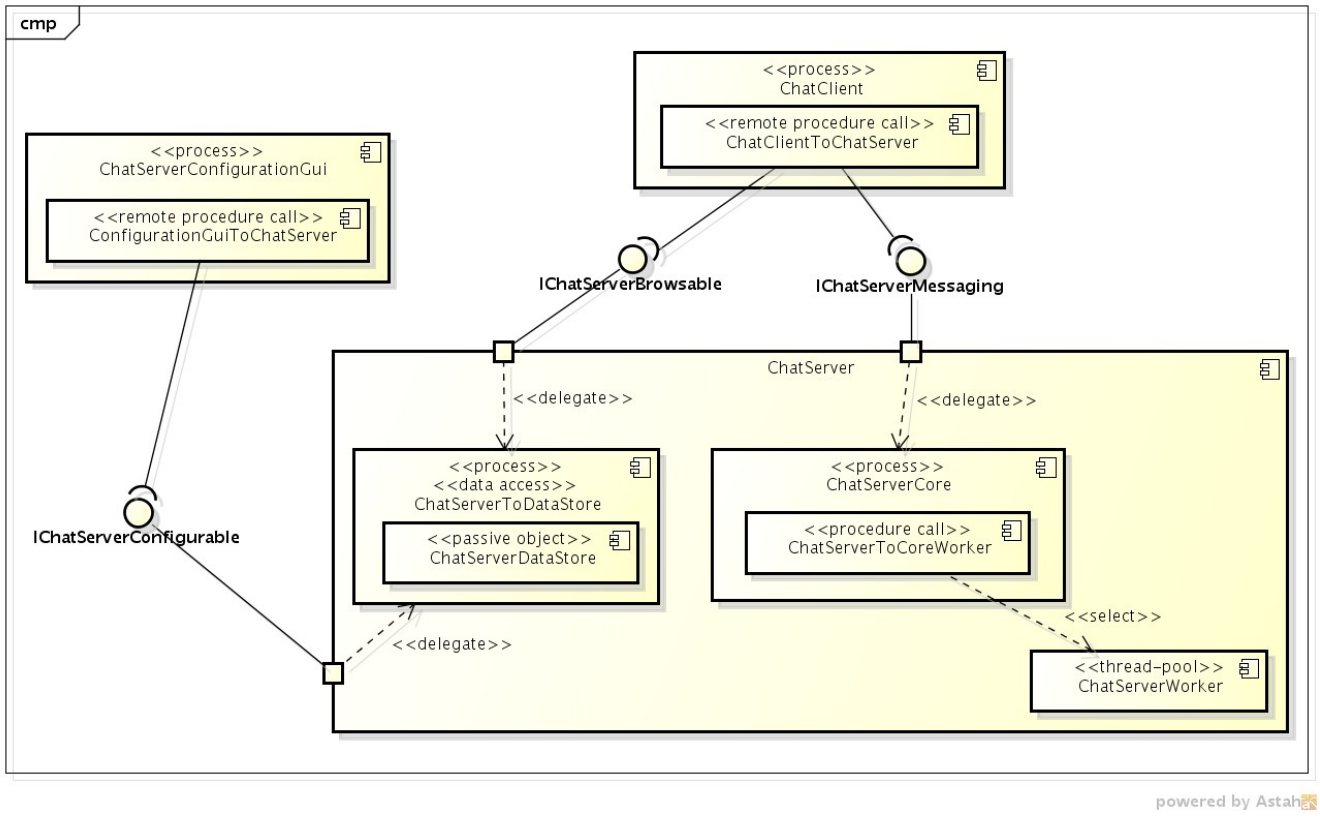


**Figura 4:** exemplo de modelo de concorrência

O objetivo desta visão é descrever quais componentes/conectores serão objetos ativos (detentores de *thread* própria) e quais serão passivos (aqueles que dependem do controle de outro componente para funcionar). No modelo de concorrência os elementos (componentes e conectores) da visão estrutural devem ser encapsulados em elementos arquiteturais que representarão *threads*, processos, etc. Um exercício interessante é identificar quantas *threads* são necessárias para que o componente execute sem bloqueio. No caso do *ChatServerCore* a utilização de uma única *thread* geraria impactos consideráveis no tempo de resposta de cada cliente. Por isso, ao receber um pedido de conexão, o *ChatServerCore* seleciona uma *thread* do *pool* de *ChatServerWorkers* (ou cria uma nova instância de *ChatServerWorker* numa solução menos escalável) e dedica esta *thread* ao cliente que solicitou a conexão. Com isso o *ChatServerCore* pode retornar rapidamente ao atendimento de novos clientes, melhorando o tempo de resposta geral. No caso do *ChatServerConfigurationGui*, visto que as operações de leitura e escrita no *ChatServerDataStore* são geralmente rápidas, uma única *thread* será utilizada na aplicação. O mesmo vale para o *ChatClient*.

## PARTE IV – Diretrizes para Elaboração de Textos Científicos

A elaboração de redações em língua portuguesa é uma atividade que envolve uma série de aspectos, dentre eles: conhecimento técnico sobre o tema em questão, domínio e aplicação intencional dos mecanismos e construtores do idioma, bem como aspectos estilísticos particulares de cada autor.

No desenvolvimento de textos científicos (artigos, relatórios, monografias, teses, etc), entretanto, certos aspectos são considerados característicos e fundamentais. O objetivo é a produção de documentos fundamentados, com pouca ambiguidade, relativamente autocontidos, contextuais e autocríticos.

Neste seção apresenta-se as diretrizes gerais, erros comuns e *checklists* para produção de textos científicos. As diretrizes estão classificadas em dois grupos: essenciais e estilísticas. As diretrizes essenciais apresentam as normas e construtores obrigatórios na produção bibliográfica científica. As estilísticas compreendem sugestões de um determinado autor, julgadas por ele como mecanismos indutores das qualidades acima descritas como desejadas.

**Diretrizes Essenciais:**

- O título do artigo/monografia deve ser condizente com o que é apresentado no texto: nem mais nem menos.
- O resumo deve apresentar, sucintamente, todas as fases do projeto: o contexto no qual está inserido, problema, solução e avaliações.
- Tente antecipar a estrutura geral do texto, organize previamente seus pensamentos e ideias para identificar a melhor forma de exposição. O sequenciamento do conteúdo é muito importante: *first things first*. A condução das ideias é igualmente importante: evite saltos temáticos entre um parágrafo e outro. Pense em como o leitor gostaria de ler o seu texto.
- A Introdução é sua única chance de cativar o leitor. Evite o detalhe antecipado (ir direto demais ao assunto), mas também não subestime o leitor contextualizando demais. A Introdução é um desenvolvimento do resumo apresentado, elaborando o contexto, apresentando o problema em questão e uma visão geral da solução proposta. O último parágrafo da Introdução deve apresentar a estrutura do restante do artigo.
- A Fundamentação Teórica deve apresentar o estritamente necessário para a compreensão do restante do texto, nem mais nem menos. Definições, conceitos, jargões e técnicas devem ser suportadas por referências consolidadas.
- Escolha sempre a melhor referência existente para a cobertura do tópico em questão. Pode-se classificar as referências em ordem decrescente de consolidação (importância) em: livros, periódicos, artigos em conferências, relatórios técnicos e web sites. Note que utilizar web sites, tais como Wikipedia, é a sua última opção.
- Seja sempre consistente no seu texto. Encontre um bom balanço entre consistência e riqueza de vocabulário. Utilizar, por exemplo, o termo "*middleware*" em uma parte do texto e "*framework*" em outra - somente para evitar a repetição de expressões - deixa de ser justificado a partir do momento em que começa a confundir o leitor. Por outro lado, "sistema", "aplicação" e "*software*" podem ser intercambiáveis sem prejuízo (desde que o seu texto não discorra sobre a diferença entre sistema, aplicação e software).
- Termos da língua inglesa devem sempre ser escritos em itálico. Isto inclui palavras tais como *software* e *hardware*, pois apesar de estarem presentes nos dicionários da língua portuguesa, não foram "aportuguesados" (não escrevemos sófituer, porém shampoo foi aportuguesado para xampú). Polêmico ?
- Evite frases longas demais, isso prejudica a compreensão e cansa o leitor. Você pode combinar com o seu professor o número máximo de linhas de cada frase. Uma boa regra estilística é intercalar frases maiores com frases menores. Exemplo ruim: "*o desenvolvimento de sistemas complexos geralmente é custoso, pois gerenciar a complexidade requer a aplicação de diversas técnicas, incluindo os padrões de projeto, pois promovem a separação de interesses e favorecem a extensibilidade, geralmente prejudicada nas atividades de evolução*". Frase reformulada: "*O desenvolvimento efetivo de sistemas complexos demanda a utilização de um conjunto de técnicas e ferramentas. Sua correta aplicação, entretanto, não é trivial. Os Padrões de Projeto se destacam como uma dessas técnicas, no sentido em que promovem uma maior separação de interesses e favorecimento da extensibilidade. Consequentemente, a evolução procede de forma menos custosa*".
- Toda sentença do seu texto deve ser bem formada. Cuidado com as frases incompletas, por exemplo "*A economia mundial, em consequência das constantes crises que fragilizam os mercados internos, cada vez mais dependentes de capitais estrangeiros.*"

- Preze religiosamente pelas concordâncias. Cuidado com as concordâncias entre termos distantes na sentença e que podem ser ocultados por um aposto que induz o erro. Exemplo incorreto: "*A Engenharia de Software e Computação Gráfica, duas áreas importantes e extensivamente presente no mercado, constitui pilar promissor no desenvolvimento dos sistemas da próxima geração*". Você conseque encontrar os quatro erros ?
- Cuidado com os usos incorretos do infinitivo, por exemplo, "*O carro estar quebrado*". O contrário também vale: "Você pode dá uma olhada !" :)
- Nunca utilize expressões figuradas ou conotativas sem uma clara justificativa didática para isso. Exemplos: "*colocou-se o software para rodar*" (pneu de carro ? Use "*executar*"), "*a técnica XXX foi utilizada para enxugar o código*" (quem molhou ? Use "*simplificar*").
- Evite afirmações pessoais, passionais e categóricas. Exemplos ruins: "*O algoritmo de ordenação XXX é a melhor solução para este domínio de aplicação*" (afirmação categórica - você conhece todas as soluções existentes no mundo ? Use "*... uma das promissoras soluções ...*". Outro exemplo: "*A ferramenta apresentou um desempenho fantástico e certamente a utilizarei em meus futuros trabalhos científicos*". Afirmar que o desempenho foi "*fantástico*" denota afirmação passional e "*... utilizarei em meus futuros trabalhos ...*" é uma afirmação pessoal, afinal qual a relevância científica disso para o texto ?

**Diretrizes Estilísticas**

- O texto deve ter um caráter impessoal. Diferente dos textos produzidos na língua inglesa, onde é comum encontrarmos expressões do tipo "*... we designed ... we developed*", geralmente não se utiliza a primeira pessoa em textos em português. Prefira "*Foram analisados ...*" em vez de "*Eu analisei*", "*Pode-se notar ...*" em vez de "*Pudemos notar*".
- As sentenças que descrevem as suas realizações devem estar todas no passado: "*projetou-se*", "*foi implementado*", "*avaliou-se*", etc. Uma exceção são os pré-projetos, onde você diz o que irá fazer no futuro.
- Os números geralmente são escritos por extenso, com exceção daqueles que iriam requerer sentenças maiores. Use: "*... utilizou-se cinco máquinas para o experimento ...*" em vez de: "*... utilizou-se 5 máquinas para o experimento ...*". Porém utilize "*... foram realizadas 256 repetições do experimento ...*" em vez de "*... foram realizadas duzentos e cinquenta e seis repetições do experimento ...*"
- Cada item de uma itemização deve terminar com ";" (se curto) ou "." (se mais extenso). A justificativa é que itens curtos seriam integrantes de uma mesma frase, "itemizada" somente para destacar os elementos constituintes. Exemplos:

> *Os principais representantes das transformadas relacionadas a Fourier são:*
>
> - *Transformada de Fourier;*
> - *Série de Fourier;*
> - *Transformada de Fourier de Tempo Discreto; e*
> - *Transformada Discreta de Fourier.*

Entretanto,

> *Christopher Alexander em seus livros Notes on the Synthesis of Form, The Timeless Way of Building e A Pattern Language, estabelece que um padrão deve ter, idealmente, as seguintes características:*
>
> - *Encapsulamento: um padrão encapsula um problema/solução bem definida. Ele deve ser independente, específico e formulado de maneira a ficar claro onde ele se aplica.*
> - *Generalidade: todo padrão deve permitir a construção de outras realizações a partir deste padrão.*
> - *Equilíbrio: quando um padrão é utilizado em uma aplicação, o equilíbrio dá a razão, relacionada com cada uma das restrições envolvidas, para cada passo do projeto. Uma análise racional que envolva uma abstração de dados empíricos, uma observação da aplicação de padrões em artefatos tradicionais, uma série convincente de exemplos e uma análise de soluções ruins ou fracassadas pode ser a forma de encontrar este equilíbrio.*

***Checklist* (indispensável – ver barema):**

1. Utilizei o corretor ortográfico antes de enviar o texto para o professor ?
2. O meu texto está livre de erros de concordância ?
3. Estou usando resumos estruturados ? O ideal é ter uma sentença para cada um dos seguintes itens: background, objetivos, método, resultados e conclusão.
4. O meu texto está livre de expressões figuradas, coloquiais ou passionais ? Exemplos de frases ruins: "colocou-se o software para rodar", "a técnica XXX foi utilizada para enxugar o código", "A ferramenta apresentou um desempenho fantástico"
5. O último parágrafo da introdução apresenta a estrutura restante do artigo ?
6. O meu texto está livre de expressões em primeira pessoa ? Exemplos: em vez de "Eu/Nós analisei/analisamos", use "Foram analisados". Em vez de "Iremos mostrar", se "Apresentar-se-á".
7. Exceto para o caso de *position paper/work in progress*, pré-projetos e trabalhos futuros, todos os tempos verbais do meu artigo estão no passado ou presente ? Exemplo: "Este artigo apresenta", em vez de "Este artigo irá apresentar".
8. O meu texto NÃO apresenta espaços antes dos sinais de vírgula e de ponto ? O meu texto APRESENTA espaço antes de '(' e referências ? O meu texto NÃO apresenta (ponto-e-)vírgula antes do 'e' ?
9. Todos (todos significa TODOS) os termos em inglês estão em itálico ?
10. Todos os parágrafos contêm mais de uma frase ?
11. Todas as seções contêm mais de um parágrafo ?
12. Todas as frases possuem no máximo 5 linhas de texto ?
13. Todas as siglas foram explicadas na primeira vez em que aparecem no texto ? Exceções: siglas óbvias tais como API, GUI, etc.
14. Todos os termos, tecnologias, *frameworks*, *middlewares*, etc possuem referências ? Estas referências são as melhores possíveis ? Ordem de preferência (melhor para pior): artigo recente -> livro -> *website*.
15. Estou usando letra minúscula após o dois-pontos ?

**Observações Finais**

Dúvidas devem ser enviadas para sandroandrade@ifba.edu.br
Recomenda-se fortemente a execução de mais de uma iteração com o professor antes da entrega final

## PARTE V – BAREMA PARA AVALIAÇÃO DE TRABALHO PRÁTICO

| **Equipe:** | |
|---|---|
| | |

| **ESTRUTURA E LINGUAGEM** | | |
|---|---|---|
| **Item (valor)** | **Nota** | **Comentários** |
| Organização geral do texto, distribuição das seções e parágrafos, formatação (1,0) | | |
| Ortografia, concordância, parcimônia, clareza (1,0) | | |

| **CONTEÚDO TÉCNICO** | | |
|---|---|---|
| **Item (valor)** | **Nota** | **Comentários** |
| Apresentação inicial e contextualização do tema (0,5) | | |
| Apresentação dos requisitos funcionais e não-funcionais (1,0) | | |
| Uso correto de linguagem técnica (jargões, metodologias, ferramentas, etc) (0,5) | | |
| Apresentação e explicação das visões arquiteturais (2,0) | | |
| Apresentação e explicação do projeto detalhado (2,0) | | |
| Discussão sobre a implementação e como a arquitetura induz o atendimento dos requisitos não-funcionais (2,0) | | |

| **PENALIDADES** | | |
|---|---|---|
| **Item (valor)** | **Nota** | **Comentários** |
| Violação de item de checklist (0,1 por ocorrência) | | |
| Atraso na entrega (1,0 por dia) | | |

| **Nota Final** | | |
|---|---|---|

| **COMENTÁRIOS GERAIS** |
|---|
| |