Laboratório 1.2 - Oficina
Bate-papo simples

Objetivo

Implementar um sistema de bate-papo simples via TCP/IP.

Instruções

Considerando os requisitos listados, implemente o sistema.

Discussão

Coesão é um atributo de qualidade que pode ser mensurado em diversos níveis de abstração. Podemos dizer que um sistema é coeso que suas funcionalidades servem a um propósito bem definido. Também podemos avaliar a coesão de um módulo ou classe de um sistema, uma boa coesão nesse nível significa que o módulo/classe possui uma função bem definida e específica dentro do conjunto de funcionalidades que o sistema possui. Por fim, podemos avaliar a coesão no nível de método, se o método for coeso significa que o mesmo executa um bem definido e único procedimento.

Ao buscar coesão, é natural dividir o sistema em várias sub-partes coesas. As partes do sistema precisam conhecer umas às outras a fim de colaborar em tempo de execução. Quando uma parte A do sistema faz referência (chama um método, guarda um tipo de variável, recebe um tipo por parâmetro) a outra parte B, dizemos que A está acoplado a B. Isso implica que se algo for mudado em B, é possível que A seja impactado.

Acoplamento deve ser gerenciado em um projeto de software. Imagine um sistema onde todas as partes integrantes fazem referências umas às outras. Em tal situação, uma pequena mudança em uma parte específica teria potencial para disparar mudanças em todas as demais partes do sistema. Ao busca baixo acoplamento, busca-se diminuir o impacto de mudanças no sistema quando houverem mudanças.

Há um slogan muito comum usado na comunidade de métodos ágeis que diz "*Embrace change*", no sentido de abraçar, acolher a mudança. Esse slogan foca na questão cultural, já que há naturalmente resistências para realizar mudanças em sistemas já prontos.

Do ponto de vista técnico de design de software, o que é importa é "*Prepare to change, like it or not*". Ou seja, prepare-se para a mudança (é inevitável), goste ou não.

Descrição

1. Sistema de bate-papo simples

- Sistema composto por aplicação cliente e servidor.
- Aplicação servidora recebe por parâmetro de linha de comando o número da porta que deve escutar.
- Aplicação cliente recebe por parâmetro de linha de comando o endereço IP e número da porta que deve conectar.
- Requisitos:
  - O sistema aceita troca de texto livre entre cliente e servidor.
  - Quando o cliente se conecta ao servidor, em ambas aplicações deve ser mostrada uma mensagem indicando “Conexão estabelecida!”
  - Sempre que na aplicação cliente algo for escrito e inserido um “Enter”, a mensagem é enviada ao servidor.
  - Do lado do servidor, as mensagens do cliente devem ser exibidas com quebra de linha entre as mesmas.
  - Na aplicação do servidor, também deve ser possível enviar mensagens para o cliente. O envio e exibição das mensagens segue o mesmo padrão citado acima.
  - Quando o cliente desconecta, o servidor deve exibir uma mensagem informando a desconexão.

2. Modele as classes do sistema

- A partir dos requisitos, elenque os interesses a serem tratados no desenvolvimento.
- Determine quais interesses devem ser tratados na aplicação cliente e na aplicação servidora.
- Modele as classes de cada aplicação, separando os interesses.
- Revise sua modelagem com um colega ou o professor.

3. Implemente o sistema, usando ciclo de desenvolvimento iterativo e incremental.

- Não busque desenvolver uma das aplicações por completo, antes de desenvolver a outra. Desenvolva ambas em conjunto, evoluindo cliente e servidor aos poucos e incorporando funcionalidades de forma incremental.