Critérios de qualidade que pautam a lista de verificação do Labutil (Bastien e Scapin, 1993)

Os critérios de qualidade se constituem em um refinamento das regras de Shneiderman.

1. Presteza

Verifique se o sistema informa e conduz o usuário durante a interação

Isto envolve:

- meios utilizados para levar o usuário a realizar determinadas ações, como por exemplo, entrada de dados;
- informações que permitem ao usuário identificar o estado ou contexto em que ele se encontra;
- mecanismos que permitem ao usuário conhecer as alternativas, em termos de ações, conforme o estado do sistema ou contexto em que ele se encontra;
- ferramentas de auxílio e seu modo de acesso.

Exemplos:

- Fornecer rótulo não-ambíguo para cada campo de formulário (“endereço” é ambíguo, “rua” não);
- Dirigir a entrada de dados indicando o formato adequado e os valores aceitáveis (Data: _ _/_ _/_ _);
- Exibir as unidades associadas aos dados (cm, m,...);
- Indicar a não-necessidade de zeros à esquerda, de separadores entre número e dígito verificador e outros dados eventualmente desnecessários;
- Indicar todas as informações sobre o estado da interação;
- Indicar o tamanho do campo, quando ele é limitado (20 caracteres, por exemplo);
- Dar título a cada janela;
- Fornecer ajuda *online* e orientação para cada ação disponível ao usuário;
- Realização de ações deve ser aparente, especialmente no caso de esta ser diferente do padrão na plataforma do sistema e/ou em outros sistemas da mesma aplicação (contraexemplo: envio de mensagem por ENTER no Hotmail)

2. **Agrupamento espacial por significado**

Verifique se a distribuição espacial dos itens traduz as relações de significado entre as informações e funções

Isto envolve:

- posicionamento relativo dos itens, estabelecido para indicar se eles pertencem ou não a uma dada classe, ou, ainda, para indicar diferenças entre classes;
- posicionamento relativo dos itens dentro de uma classe.

Exemplos:

- Organizar as opções de uma interface por menus em função dos objetos aos quais elas se aplicam (objetos-ações ou ações-objetos);
- Definir ordem de apresentação relevante (ordem alfabética, freqüência de uso,...);

- Separar grupos de funções por espaçamento ou linhas.

3. Agrupamento visual por significado

Verifique se as características visuais dos itens são exploradas como meio de transmitir associações e diferenças

Isto envolve:

- características visuais (formato, cor e outros) que indicam se itens pertencem ou não a uma dada classe, ou que indicam ainda distinções entre classes diferentes ou entre itens de uma mesma classe

Exemplos:

- Fazer distinção visual clara de áreas que têm diferentes funções (área de entrada de comandos, de *feedback*, de mensagens, ...);
- Fazer uma distinção visual clara dos campos.

4. Feedback

Avalie a qualidade do *feedback* (retorno) imediato às ações do usuário

Isto envolve:

- respostas do sistema às ações do usuário (desde o simples pressionar de uma tecla até a emissão de uma lista de comandos) com informações sobre a transação solicitada e seu resultado

Exemplos:

- Exibir todas as entradas, com exceção dos dados sigilosos. Mesmo neste caso, cada entrada deve produzir um *feedback* perceptível (como “*”);
- Seguindo a interrupção pelo usuário de um processamento de dados, mostrar mensagem garantindo ao usuário que o sistema voltou ao seu estado prévio (crucial em sistemas de transações);
- Para processos demorados, mostrar informações sobre o estado do processamento (barrinha exibindo percentual) com mensagens bem-definidas das ações sendo executadas (“leitura concluída”, por exemplo).

5. Legibilidade

Verifique a legibilidade das informações apresentadas nas telas do sistema

Isto envolve:

- características visuais das informações apresentadas que possam facilitar ou dificultar a leitura dessa informação (exemplos: brilho do caractere, contraste letra-fundo, tamanho da fonte, espaçamento entre palavras, entre linhas e entre parágrafos, comprimento da linha, entre outros).

Exemplos (heurísticas):

- Títulos devem ser centralizados;
- Rótulos devem estar em letras maiúsculas;
- Cursores devem se apresentar distintos de outros itens possíveis;

- Em espaço limitado, preferir poucas linhas longas (ao menos 50 caracteres) ao invés de muitas linhas curtas;
- Usar justificação à direita apenas se puder ser mantido o espaçamento proporcional e constante entre as palavras, de forma a evitar falsas associações;
- Ao exibir um texto, as palavras devem ser mantidas intatas, com o mínimo de hífens e separadores.

6. Concisão

Verifique o tamanho dos códigos e termos apresentados, exigidos e introduzidos no sistema

Isto envolve:

- carga preceptiva e cognitiva de saídas e entradas individuais

Exemplos:

- Para entradas numéricas, zeros à esquerda não devem ser necessários; para números compostos, separadores idem. Isto deve ficar aparente;
- Para códigos muito longos (maiores que 4 ou 5 caracteres) usar mnemônicos ou abreviaturas (contradição);
- Permitir ao usuário entradas de dados sucintas. Deixar isso aparente;
- Quando a unidade associada a um dado é conhecida, deve ser exibida pelo sistema ao lado do campo, ao invés de exigir que o usuário a digite.

7. Ações mínimas

Verifique a granularidade das ações e a extensão dos diálogos estabelecidos para a realização dos objetivos do usuário

Isto envolve:

- carga de trabalho em relação ao número de ações necessárias à realização de uma tarefa (problema de esforço de especificação X poder expressivo e especificidade da linguagem)

Exemplos:

- O número de passos necessários para se fazer uma seleção por menus deve ser minimizado (compromisso);
- O usuário não deve ser obrigado a entrar com dados que o sistema poderia gerar;
- Valores *default* e correntes devem sempre ser exibidos no processo de atualização de dados;
- Quando várias páginas de informação estiverem envolvidas, deve-se permitir ir diretamente a páginas específicas sem necessidade de passar seqüencialmente pelas intermediárias (hipertexto).

8. Densidade informacional

Avalie a densidade informacional das telas apresentadas pelo sistema

Isto envolve:

- carga de trabalho do usuário de um ponto de vista perceptivo e cognitivo, com relação ao conjunto total de itens de informação apresentados simultaneamente nas telas do sistema

Exemplos:

- Em qualquer transação, fornecer somente dados que sejam necessários e diretamente utilizáveis pelo usuário no estado do sistema;
- Os dados não devem necessitar de tradução entre unidades. Devem ser sempre usadas as unidades naturais na cultura do usuário;
- Linguagens de consulta devem usar o mínimo possível de quantificadores na formulação de consultas. Quando presentes, significado deve ficar bem aparente. Em particular, evitar negação pela ambigüidade semântica na linguagem natural;
- Deve-se evitar exigir do usuário que ele lembre de dados exatos de uma tela para outra.

9. **Ações explícitas e** 10. **Controle do sistema pelo usuário**

Verifique se é o usuário quem comanda as ações do sistema. Avalie as possibilidades de o usuário controlar o encadeamento e a realização de ações

Isto envolve:

- relação entre o processamento realizado pelo computador e as ações do usuário. O processamento só deve ocorrer quando o usuário solicita ações;
- recomendação de que os usuários estejam sempre no controle do processamento do sistema (possibilidade de interromper, cancelar, suspender e retomar tarefas deve ser sempre possível);
- antecipação ao usuário das tarefas possíveis em cada instante.

Exemplos:

- Entradas de comandos devem ser seguidas de um ENTER;
- Sempre deve ser necessário que o usuário tecle um ENTER, um OK, ou algo do tipo explicitamente para ativar o processamento de dados digitados (compromisso com eficiência);
- Nunca deve-se disparar o processamento (por exemplo, atualizar um arquivo) como efeito colateral de uma outra ação. No caso de isto ser feito, pelo menos chamar a atenção e pedir confirmação para a realização da ação;
- Para seleção por meio de dispositivos de apontamento, a ativação deve ser feita em dois passos: a primeira ação (posicionar o cursor) deve designar a opção selecionada, e uma segunda ação (“clickar”) deve disparar o controle da ação;
- Apenas a explicitação da realização das ações deixa o usuário no controle (contraexemplo: navegação inconsciente pela ignorância do significado das diferentes ações disponíveis (“*click*” e “*double click*”).

11. Flexibilidade

Verifique se o sistema permite personalizar as apresentações e os diálogos

Isto envolve:

- meios colocados à disposição do usuário que lhe permitem personalizar a interface, a fim de levar em conta as exigências da tarefa, de sus estratégias e de seus hábitos de trabalho;
- diferentes maneiras à disposição do usuário para alcançar certo objetivo;
- capacidade da interface de se adaptar às variadas ações do usuário.

Exemplos:

- Sempre que possível usuário deve poder configurar as telas ao seu gosto;

- Quando a relevância de determinada informação em todos os contextos não puder ser determinada pelo sistema, deve-se permitir ao usuário a sua desativação temporária;
- A seqüência de entrada de dados deve poder ser modificada pelo usuário para se adaptar à ordem natural para ele;
- Quando o formato preferível para textos não for conhecido pelo sistema, deve-se proporcionar ao usuário meios de seleção do formato desejado;
- O usuário deve poder modificar os nomes de campos de entrada de dados.

12. **Experiência do usuário**

Avalie se os usuários com diferentes níveis de experiência têm iguais possibilidades de obter sucesso em seus objetivos

Exemplos:

- É conveniente prever atalhos;
- É desejável permitir que usuários experientes contornem uma série de seleções por meuns por meio da especificação de comandos ou atalhos de teclado;
- É conveniente proporcionar a possibilidade de entrada de comandos simples para usuários leigos e compostos para usuários experientes;
- É aconselhável proporcionar diferentes modos de diálogo para os diferentes grupos de usuários;
- É desejável fornecer tutorial passo a passo para usuários novatos;
- É necessário fornecer ao usuário experiente modos de contornar orientação fornecida para usuários leigos (contraexemplo: necessidade de ouvir toda a orientação em serviços telefônicos antes de discar opção desejada);
- É conveniente permitir a definição de níveis de detalhe das mensagens de erro em função do nível de conhecimento do sistema pelo usuário.

13. Proteção contra erros

Verifique se o sistema oferece as oportunidades para o usuário prevenir eventuais erros

Isto envolve:

- mecanismos empregados para detectar e prevenir os erros de entrada de dados, comandos, possíveis ações de conseqüências desastrosas e/ou de ação não passível de anulação

Exemplos:

- Rótulos dos campos devem ser protegidos, não passíveis de modificação pelo usuário (contradição);
- Idem para orientações associadas à entrada de dados;
- Opções de entrada conhecidas pelo sistema devem sempre ser exibidas, minimizando possibilidade de erros de digitação (identificação de arquivos)..

14. Mensagens de erro

Avalie a qualidade das mensagens de erro enviadas aos usuários em dificuldades

Isto envolve:

- pertinência, legibilidade e exatidão da informação dada ao usuário sobre a natureza do erro cometido (sintaxe, formato e outros) e sobre as ações a executar para corrigi-lo

Exemplos:

- Se o usuário pressiona uma tecla de função inválida, nenhuma ação deve ocorrer, a não ser uma mensagem indicando as funções apropriadas a essa etapa da interação;
- Mensagens de erro devem ser orientadas a tarefas, contextuais e não-genêricas;
- Nas mensagens de erro, devem ser utilizados termos tão específicos quanto possível;
- Mensagens de erro devem ser sucintas;
- Mensagens de erro devem informar a natureza do erro cometido (sintaxe, contexto,...), o erro exato (“espaço não é um delimitador válido“), e as ações válidas para a corrigi-lo (“delimitador deve ser ‘-‘ “);
- Estilo das mensagens de erro deve ser impessoal, não repreensivo e evitar o humor.

15. Correção de erros

Verifique as facilidades oferecidas para que o usuário possa corrigir os erros cometidos

Exemplos:

- Depois de um erro de digitação de comando ou dados deve ser dada ao usuário a possibilidade de corrigir somente a parte do comando ou dos dados errada
- É desejável identificar, no andamento do processo de entrada, os erros cometidos (isto exige processamento ao longo, e contradiz a necessidade de efetivação explícita de comandos, além de estar em compromisso com a eficiência)

16. Consistência

Avalie se é mantida uma coerência no projeto de códigos, telas e diálogos com o usuário

Isto envolve:

- forma na qual as escolhas na concepção da interface (códigos, denominações, formatos, procedimentos) são conservadas idênticas, em contextos idênticos, e diferentes, em contextos diferentes

Exemplos:

- Deve ser mantido o *layout* de telas semelhante. Em particular, deve-se manter a posição dos títulos das janelas, dos *prompts* para a entrada de comandos ou dados e das mensagens do sistema (*feedback* e erro);
- Na condução e nas mensagens de erro, é bom sempre utilizar as mesmas expressões construções de frases, para não dar lugar a falsas interpretações ou expectativas;
- Devem ser realizados procedimentos similares de acesso às opções funcionais em qualquer estágio da interação.

17. Códigos naturais

Avalie se os códigos e as denominações são naturais e significativos para o usuário do sistema

Isto envolve:

- adequação entre o objeto ou a informação apresentada ou solicitada e sua referência no contexto do usuário

Exemplos:

- Deve-se levar em consideração, ao criar títulos, rótulos de função, opções de menus e outros, expressões utilizadas no ambiente real onde o sistema é utilizado;
- Denominações padrão devem ser preferidas sobre as arbitrárias (“F/M” para masculino e feminino, ao invés de “½”);
- A não adesão a padrões deve ficar explícita. Por outro lado, deve-se evitar usar símbolos padrão para indicar significado diferente do esperado.

18. Compatibilidade

Verifique a compatibilidade do sistema com as expectativas e necessidades do usuário na execução de sua tarefa

Isto envolve:

- adequação da organização das tarefas, entradas, saídas e diálogo de uma dada aplicação às características do usuário (memória, percepção, hábitos, competências, idade, expectativas e outros);
- grau de similaridade entre diferentes ambientes e aplicações familiares ao usuário.

Exemplos:

- Orientação deve estar associada aos dados de entrada, isto é, deve ir acompanhando o processo de entrada e não ser dada de uma vez, de forma global;
- O formato dos formulários deve ser compatível com os formulários em papel (no caso da pertinência do reprojeto, treinamento é necessário);
- Funções disponíveis para o usuário devem estar organizadas da forma como ele as enxerga no mundo real (por exemplo, funções mais comuns e gerais devem estar acessíveis no primeiro nível);
- Formatos e unidades devem acompanhar a cultura onde o software será utilizado (contraexemplos: formato mês/dia/ano e polegadas no Brasil).