PROJETO PROCEDIMENTAL

Projeto de Programas – PPR0001

Introdução

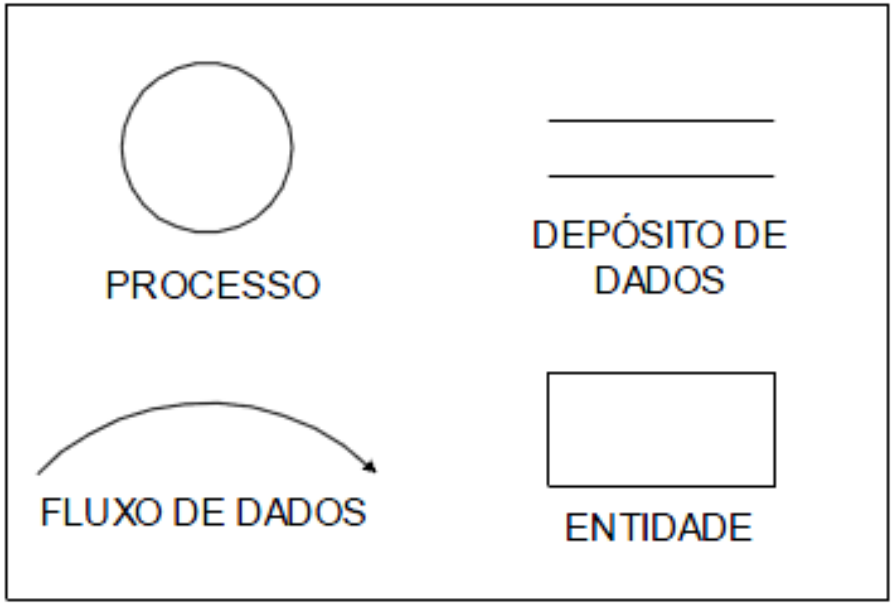
- A trípode de modelagem é composta por:
  - Modelo de Objetos: especifica a estrutura dos objetos. É importante quando muitas classes não triviais são identificadas no problema
    Diagrama Entidade Relacionamento e Diagrama de Classes
  - Modelo Funcional: especifica os resultados de um processamento sem especificar como ou quando eles serão processados. Evidencia quais dados são entradas de um processo e quais devem ser as saídas.
    Diagrama de Fluxo de Dados (DFD)
  - Modelo Dinâmico: representa a parte dinâmica do sistema, especificando os principais estados e eventos do sistema.
    Diagrama de Eventos e Diagrama de Estados

Modelagem Funcional

- O modelo funcional especifica como os valores de saída de um processamento se transformam em valores de entrada para outro processo
- Representação utilizando **Diagramas de Fluxo de Dados** (DFD)
- DFD é um gráfico que mostra o fluxo dos valores de dados desde suas origens nos objetos, através dos processos que os transformam, até seus destinos em outros objetos.

Diagrama de Fluxo de Dados

- Um DFD pode ser visto como uma rede que ilustra a circulação dos dados no interior do sistema;
- Símbolos utilizados:
- O software ASTAH permite a criação de DFDs, mas apenas na versão paga ☹



ASTAH Community é a versão gratuita da ferramenta

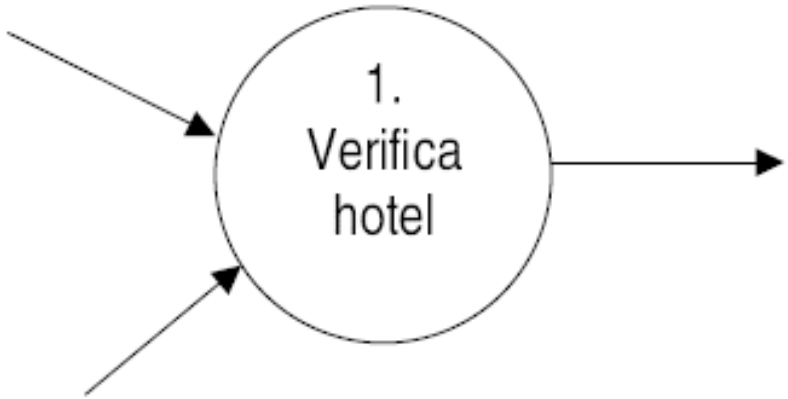
OBS: é possível usar a versão paga como estudante!

Fluxo de dados

- Representado por setas direcionadas, indicam o fluxo de um determinado conjunto de dados.
- Pode-se representar a cópia ou a subdivisão dos componentes de um dado através de um “garfo”

Processos (bolhas ou bolas)

- Transformam fluxos de dados: entrada → saída
- São identificados com um nome (e opcionalmente um número)
- Os fluxos de dados envolvidos indicam os caminhos possíveis

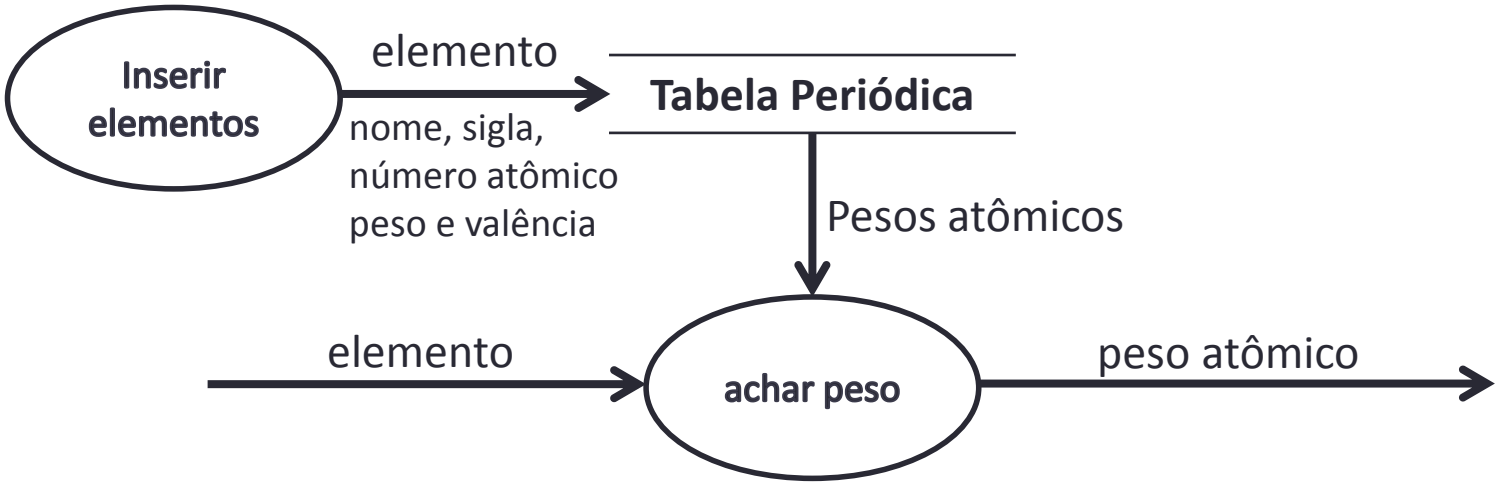


Depósito de dados (arquivo)

- São “reservatórios” para os dados existentes no sistema
  - Variável em memória, arquivos, banco de dados
- Representados por duas linhas horizontais paralelas com o nome do depósito (nome único) no meio
- Setas direcionadas a um depósito indicam a inclusão ou modificação de dados (dados entrando)
- Setas que saem de um depósito indicam a consulta ou recuperação de informação (dados saindo)

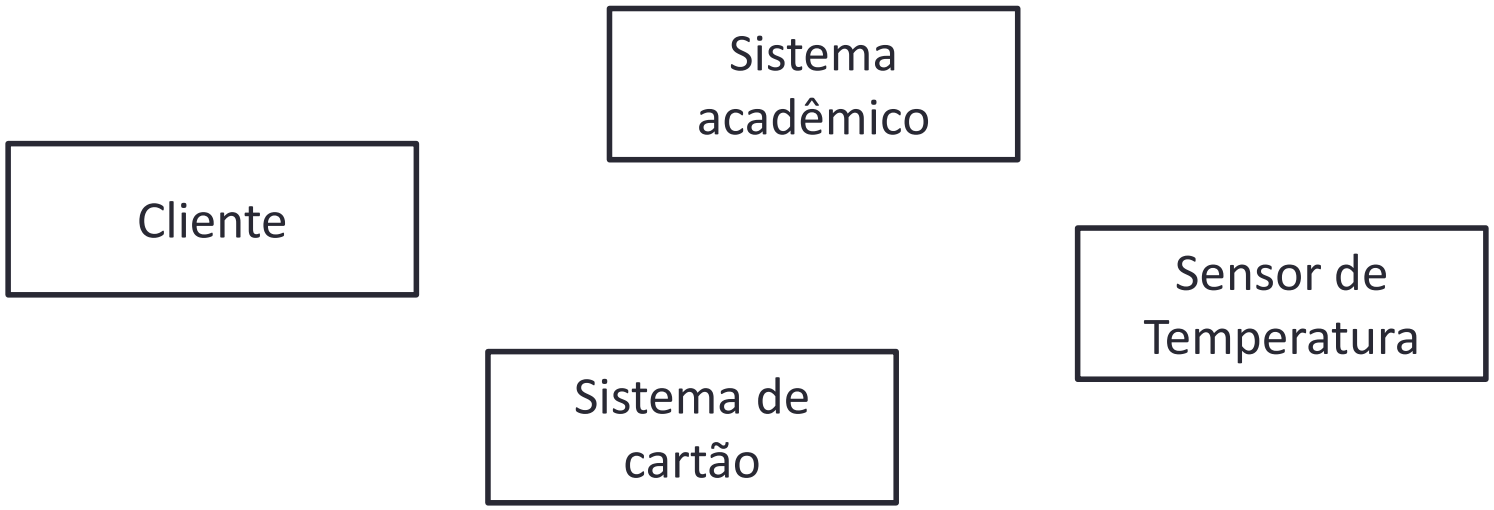
Depósito de dados (arquivo)

- Exemplo:



Entidades exteriores - atores

- Elementos que fornecem entradas e recebem saídas do sistema
  - Usuários, sistemas externos, hardware, ...
- Estão fora da fronteira do sistema (são externos ao sistema)
- Representados por retângulos



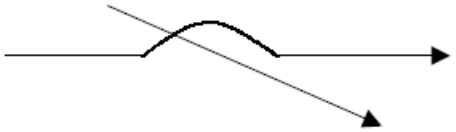
Exemplo

- Sistema de vendas:
  - Quando um cliente faz um pedido, um vendedor da loja inicialmente busca no sistema se o(s) itens pedidos estão disponíveis e informa o preço. Caso o cliente confirme a compra, ele informa a quantidade ao vendedor que realiza a baixa no sistema e emite uma nota fiscal.

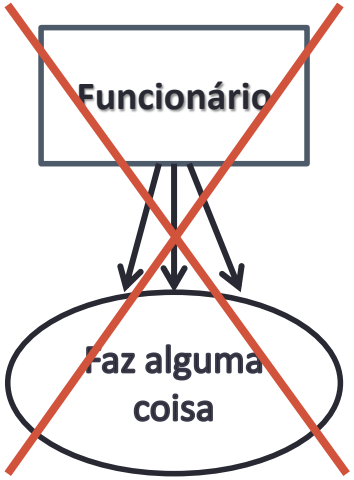
- Sistemas de reservas em um Hotel:
  - O hotel “Durma Bem” permite a reserva de quartos online. Para isso o cliente pode visualizar os quartos do hotel e sua disponibilidade. Ao selecionar um dos quartos o cliente informa a quantidade de dias da reserva e pode efetivar a reserva com dois dias de antecedência.
  - No dia referente a reserva o funcionário do hotel realiza o check-in, registrando a hora de entrada.
  - Quando o cliente deseja sair (no dia previsto ou antes), o funcionário do hotel realiza o check-out e emite a conta para o cliente.

Convenções adicionais

- **Minimizar** o cruzamento de fluxos;
- Caso ocorra cruzamento utilizar a notação:



- Repositórios e Atores podem ser desenhados mais de uma vez, mas devem ter o mesmo nome



- Não pode haver processos apenas com entradas;
- Processos só com saídas são incomuns
  - Exemplo válido: gerador de números aleatórios

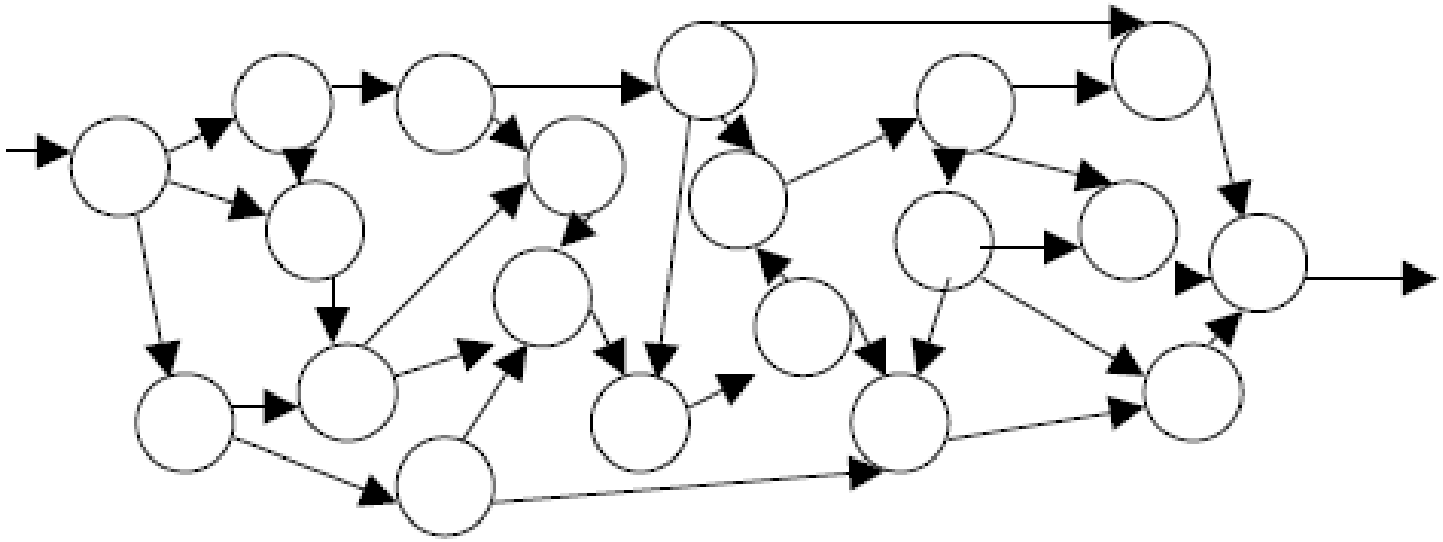
Convenções adicionais

- Omite-se detalhes de programação como verificação de erros, inicializações e finalizações.
- O uso de nomes ambíguos ou genéricos para processos e/ou fluxos de dados revelam falta de conhecimento sobre o sistema (ex. manipulação de entrada, gera saída, itens de entrada, vários dados).
- Muitos cruzamento de fluxos indicam que uma decomposição do DFD pode ser necessária.

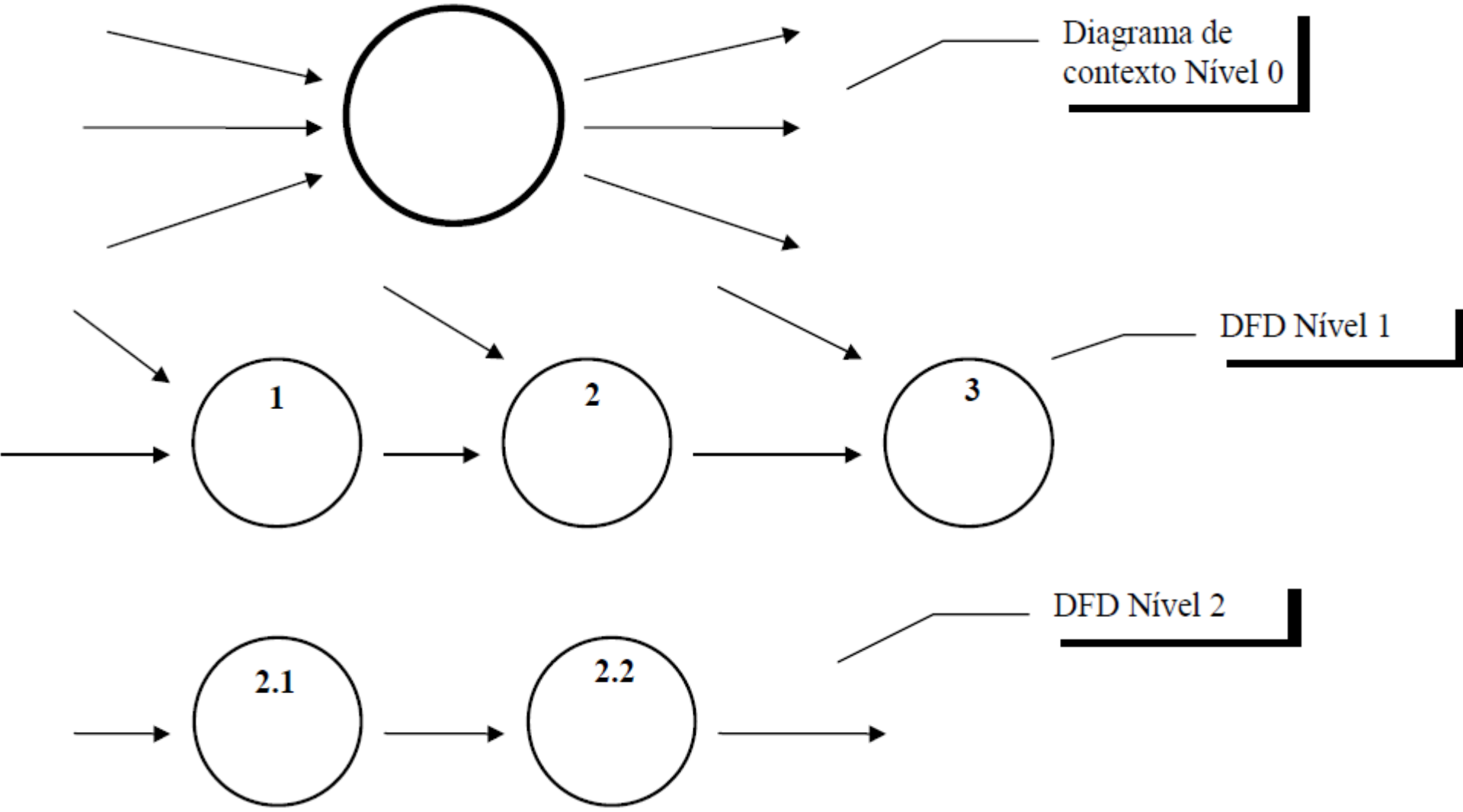
Decomposição

- Um DFD de um sistema pequeno é fácil de construir e é facilmente interpretado e entendido.

- Quanto mais complexo for o sistema modelado mais complexo o DFD poderá se tornar.



Decomposição de DFDs - Níveis



Convenções de decomposição

- Cada processo em um nível de DFD pode ser expandido para se tornar um novo DFD
- Cada processo de um nível inferior está relacionado com o nível superior e é identificado por um número composto (ex.: 2.1.3)
- Todos os fluxos de dados que entram e saem do nível superior devem aparecer no nível inferior (validação vertical)
- Recomenda-se desenhar no máximo 7 processos por DFD
- Processos muito simples não precisam ser expandidos. Costumam ser denominados de **processos primitivos** ou **primitivas funcionais**

Exemplo

- Sistema de reconhecimento de padrões
  - Escrever um script inicial para separação da base de dados original em três sub-bases: treinamento (25%), avaliação (25%) e teste (50%).
  - Depois escrever um programa que recebe três valores como entrada: o nome do diretório onde estão as bases de dados, o número de iterações de teste (**T**) e o índice da coluna que identifica a classe (a qual se está reconhecendo padrões).
  - O programa deve fazer a leitura das três bases de dados em **csv** (*comma separated values*). Então o programa utilizará a base de treinamento e de avaliação para descobrir qual é o melhor número **k** de vizinhos (1, 3, 5, 7, ...19) a ser considerado. Após descobrir o melhor valor de **k** inicia-se um processo de **T** iterações onde: testa-se cada valor da base de avaliação com a base de treino. As amostras que foram classificadas erroneamente são trocadas por amostras da mesma classe da base de treino. Durante este processo deve-se armazenar o conjunto de treino que apresentou o melhor resultado. Por fim, inicia-se a classificação da base de teste utilizando-se o melhor conjunto de treinamento encontrado. O programa deve salvar dados de precisão da predição de classes do conjunto de teste e valores de tempo de treinamento e de teste final.

Regras e Heurísticas de projeto

1. Estabelecer o contexto do DFD indicando todas as **entidades externas** do sistema;
2. Identificar todas as saídas e entradas do sistema - desenhar o diagrama de contexto (abstração geral);
3. Selecionar um ponto de partida para o projeto - desenhar os fluxos que são necessários para ir de um ponto a outro;
4. Identificar os **fluxos de dados** e **depósitos de dados**;
5. Verificar, preferencialmente com o utilizador, se o DFD representa o sistema
6. Depois de estabelecido o DFD, explodir cada processo. Repetir a decomposição até obter o detalhe suficiente

Exemplo

Gerar um DFD para um sistema acadêmico que permite o controle e gerenciamento de matricula, frequência e desempenho dos discentes e a organização das disciplinas ofertadas. O sistema acadêmico deverá permitir que os acadêmicos realizem suas matrículas nas turmas de disciplinas disponíveis, considerando restrições de pré-requisitos, número máximo de créditos (9) e limite de alunos por turma. Deverá permitir que chefes de departamento incluam novas disciplinas e novos professores, abram novas turmas para as disciplinas existentes com sala, horário, lotação máxima e professor definidos. As disciplinas só poderão ser ofertadas entre 7:30 e 12:00, e, 13:30 e 21:40, em blocos de 50 minutos por aula (hora-aula). Também deverá ser possível que professores acessem suas turmas e registrem frequência e notas para seus alunos.

Exemplo

O sistema deverá ter uma opção para finalizar o semestre, possibilitando a inclusão das notas de exame. Um aluno deverá ter frequência superior a 75% e deverá ter uma média superior a 3 para realizar exame. Caso sua nota seja maior ou igual a 7 está aprovado (desde que tenha a frequência necessária). Após a digitação das notas de exame o professor deverá finalizar a turma e o sistema mostrará o resultado final. O sistema deverá funcionar nos sistemas operacionais Windows e Linux e deverá ter seu acesso controlado por login e senha.

(adicionar atributos que considerar relevantes ao problema)

Atividade

Agora é a sua vez!

Construa um diagrama de fluxo de dados para o sistema descrito no documento que está disponível na página da disciplina.

EXEMPLO EXTRA

Sistema de Hotelaria

Exemplo – Sistema de Hotelaria

Requisitos funcionais

1. O sistema deve permitir que o Cliente faça reserva de quarto(s) em determinado(s) período(s). Neste momento, é averiguado se existe quarto disponível no período solicitado. Caso positivo, é feita a reserva do quarto e enviada a confirmação para o Cliente; para isto, são necessários os seguintes itens de informação: nome do Cliente, telefone e tipo de quarto (solteiro, casal). Caso negativo, é informado ao Cliente a não disponibilidade do quarto;

2. O sistema deve permitir o cancelamento da reserva, disponibilizando o quarto, caso o Cliente solicite;

3. O sistema deve cancelar automaticamente a reserva, caso o Cliente não compareça no hotel para hospedar-se até às 12 horas do dia da reserva, disponibilizando o quarto;

Exemplo – Sistema de Hotelaria

4. O sistema deve permitir o registro do cliente ao ocupar um quarto, reservado previamente. Caso o quarto não esteja reservado, uma mensagem de rejeição será emitida. Caso contrário, a confirmação será fornecida ao Cliente;

5. O sistema deve permitir a emissão da conta ao Cliente e a disponibilização do quarto para limpeza, no momento em que ele solicitar a sua saída;

6. O sistema deve permitir o registro do pagamento da conta. Ao efetivar o pagamento é gerado um recibo para o cliente;

7. O sistema deve permitir a disponibilização do quarto, por parte do Gerente, quando este estiver limpo.

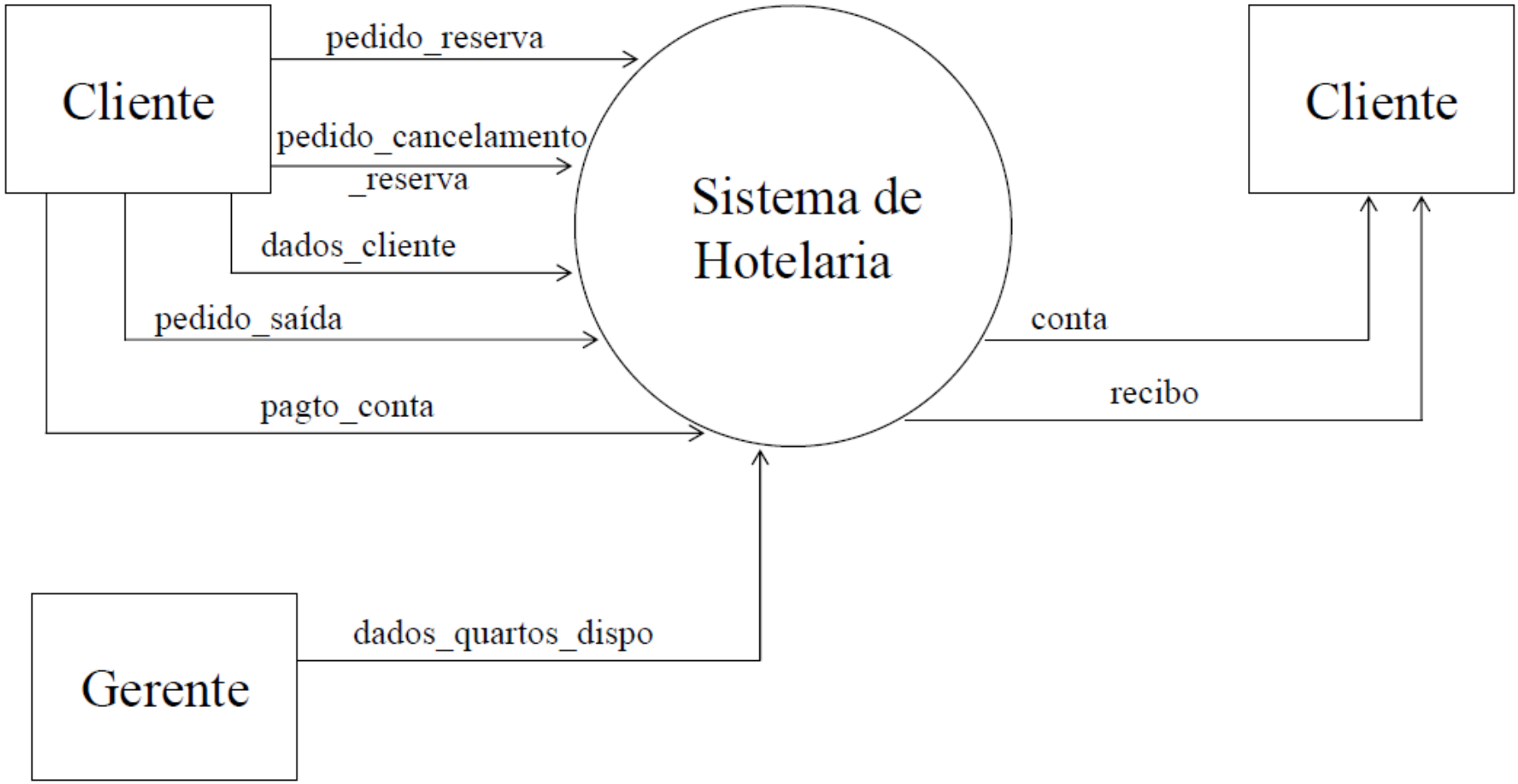
Exemplo – Sistema de Hotelaria

Eventos do Sistema

1. Cliente reserva quarto
2. Cliente cancela reserva
3. Cliente registra-se no hotel
4. Cliente solicita saída do hotel
5. Cliente paga a conta
6. Gerente disponibiliza o quarto

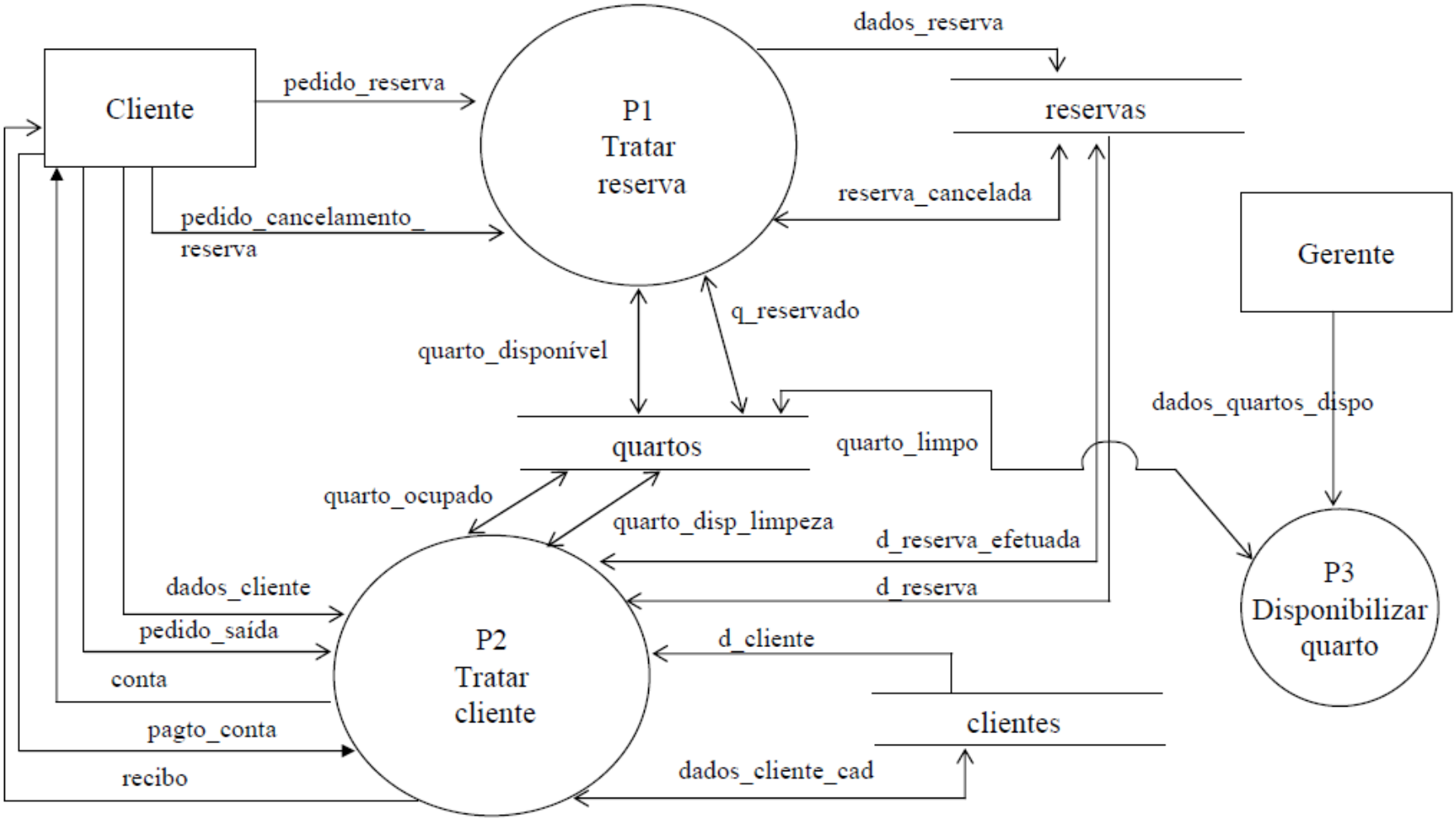
Diagrama de contexto

- Definir em uma abstração geral as entradas e saídas.



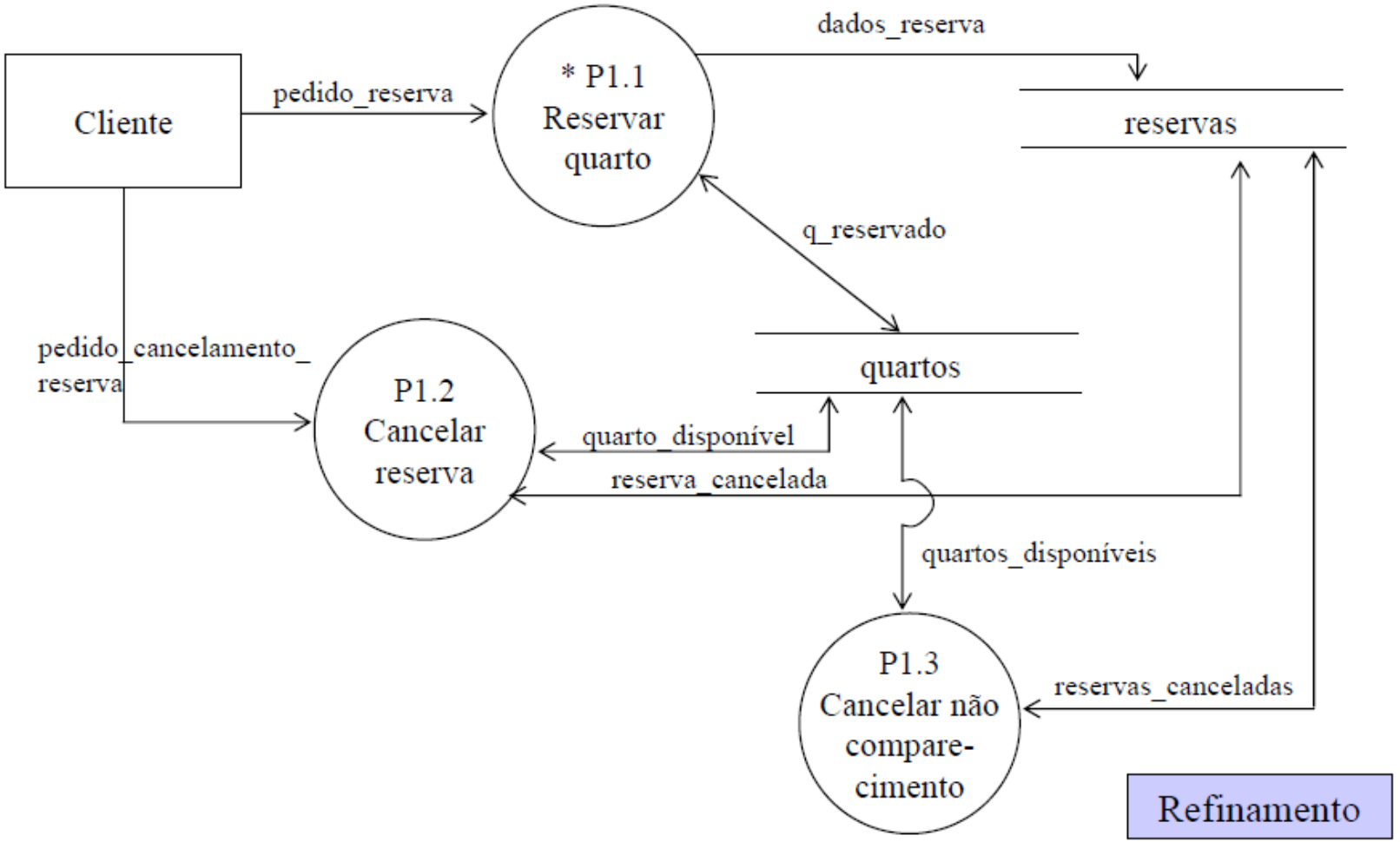
DFD – nível 0

- Visão mais específica das funcionalidades



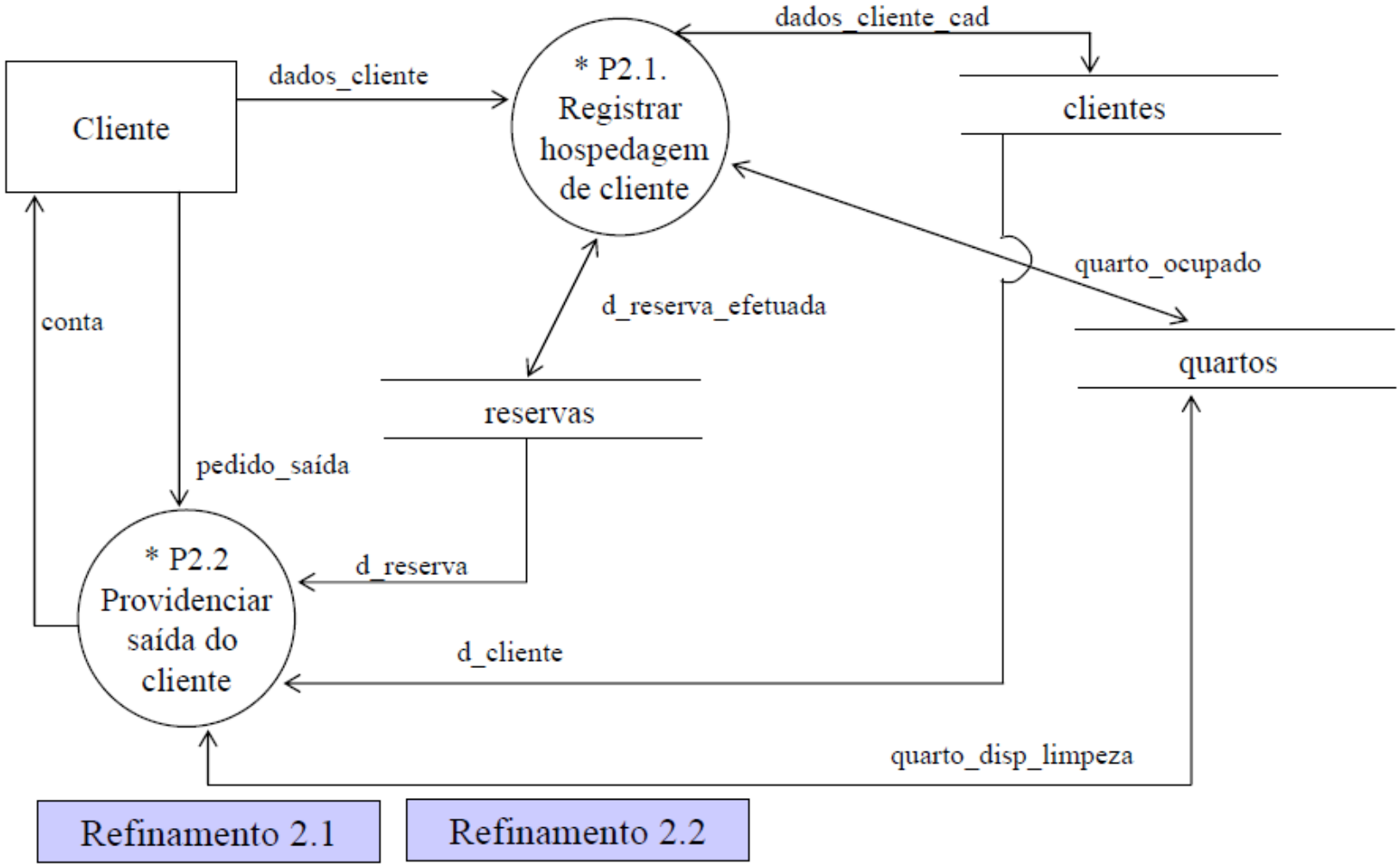
DFD – Nível 1

- Processo 1 detalhado



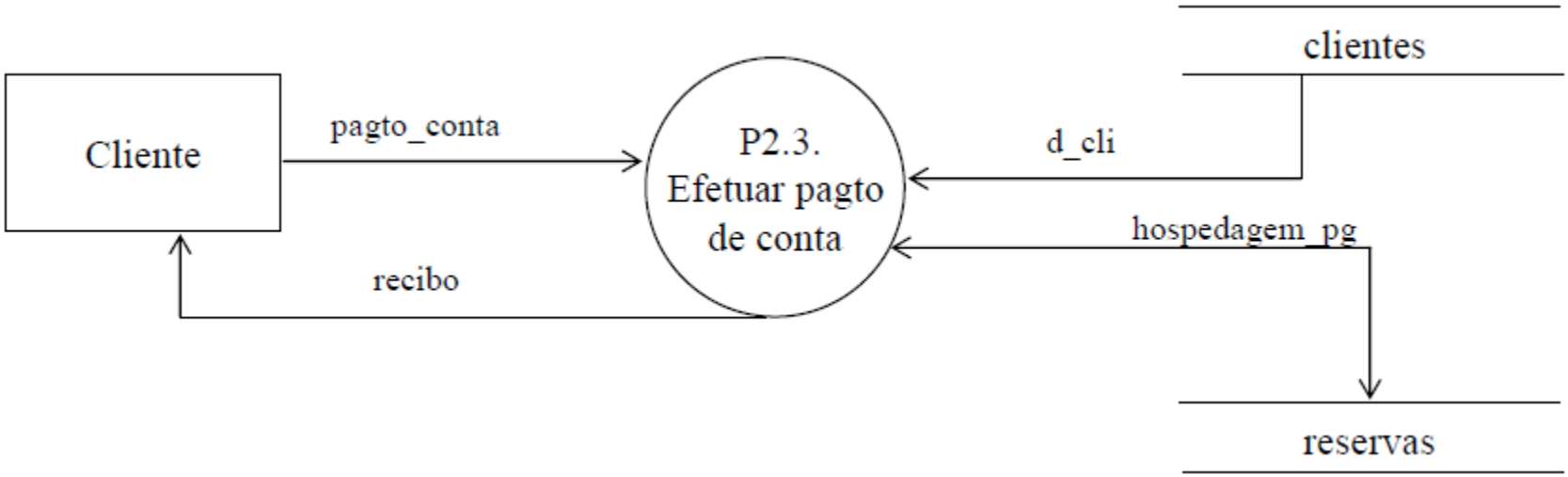
DFD – Nível 1

- Processo 2 detalhado

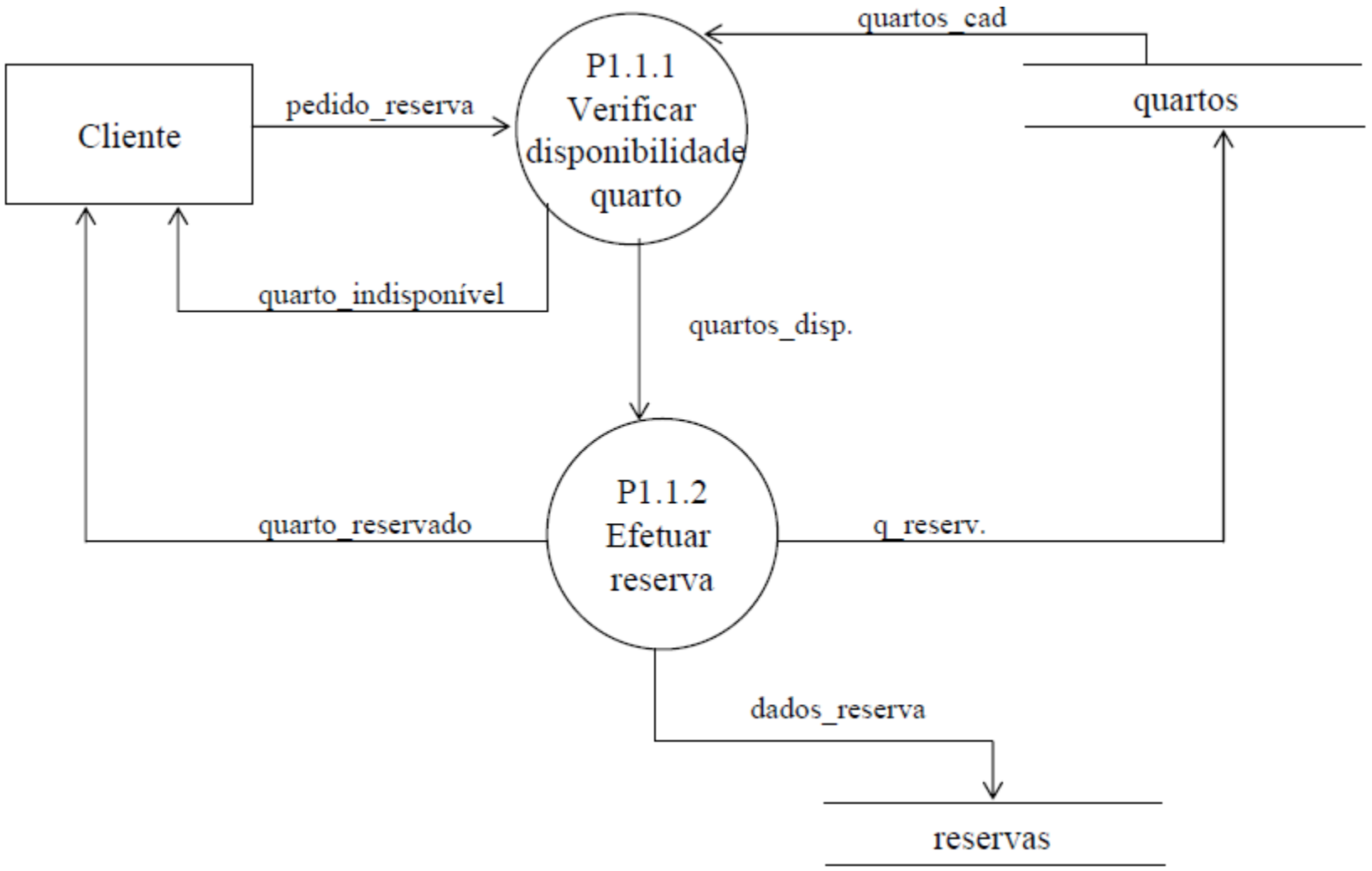


DFD – Nível 1

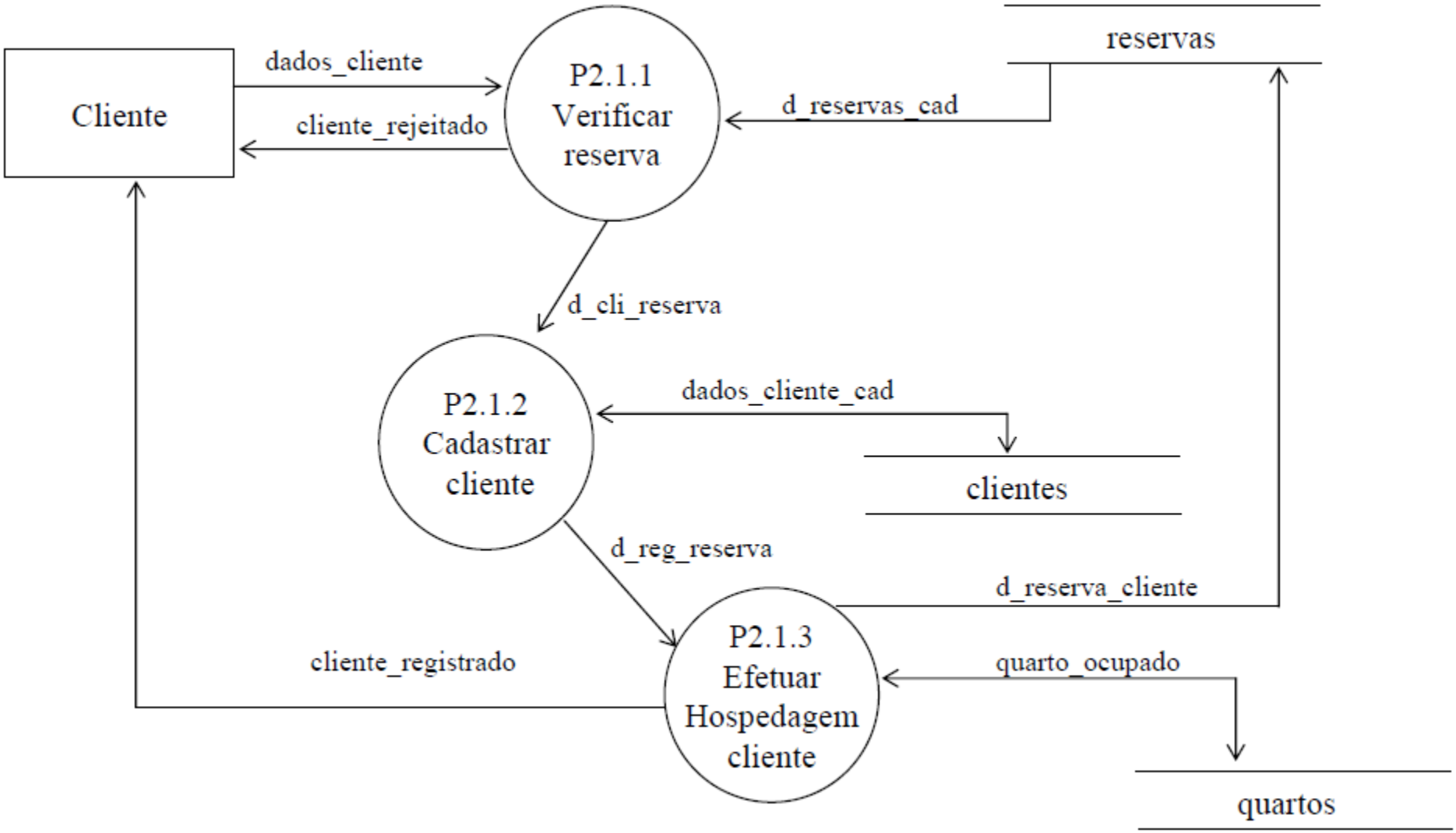
- Processo 3 detalhado



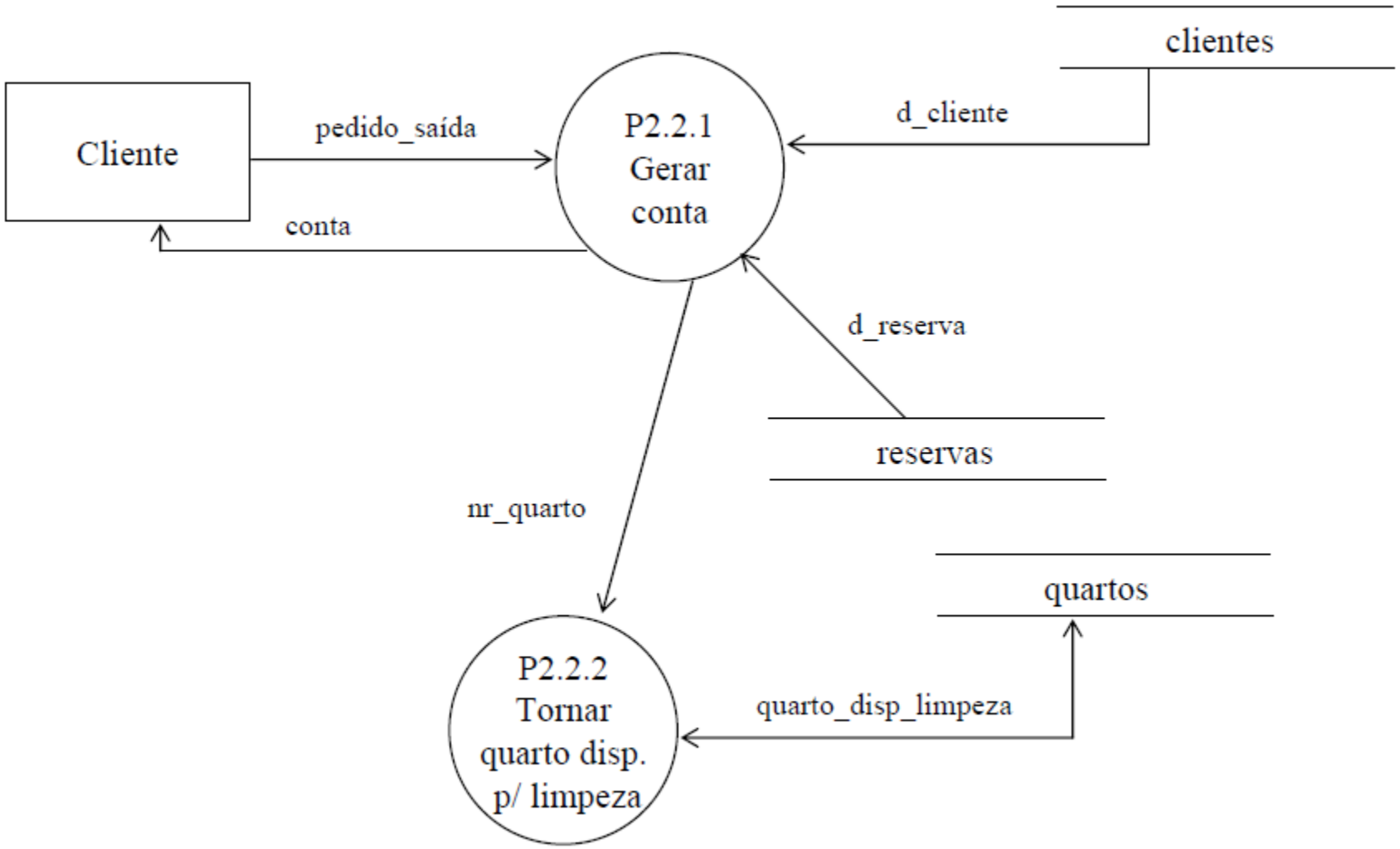
DFD – Nível 2: Refinamento de P1.1



DFD – Nível 2: Refinamento de P2.1



DFD – Nível 2: Refinamento de P2.2



Bibliografia

- **Básica:**

  BEZERRA, E. Princípios de Análise e Projetos de Sistemas com UML. Rio de Janeiro: Campus, 2003.

  PRESSMAN, R.S. Engenharia de Software. São Paulo: Makron Books, 2002.

  SOMMERVILLE, I. Engenharia de Software. São Paulo: Addison Wesley, 2003.

- **Complementar:**

  WARNIER, J. Lógica de Construção de Programas. Rio de Janeiro: Campus, 1984.

  JACKSON, M. Princípios de Projeto de Programas. Rio de Janeiro: Campus, 1988.

  PAGE-JONES, M. Projeto Estruturado de Sistemas. São Paulo: McGraw-Hill, 1988.