Projeto Final:

Modelagem de Software

Integrantes:

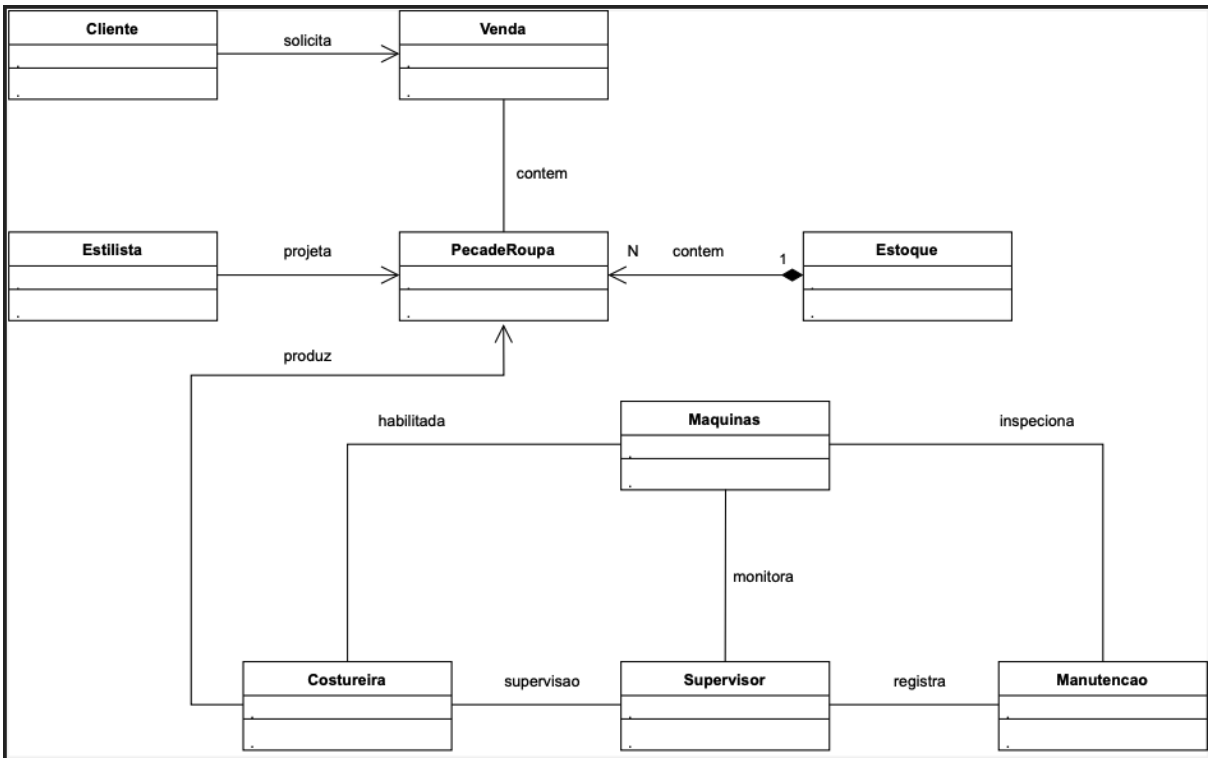
Guilherme Dias Cardoso Silva;

Guilherme Rafael Cerqueira Dias;

Luis Henrique Cardoso Palhares;

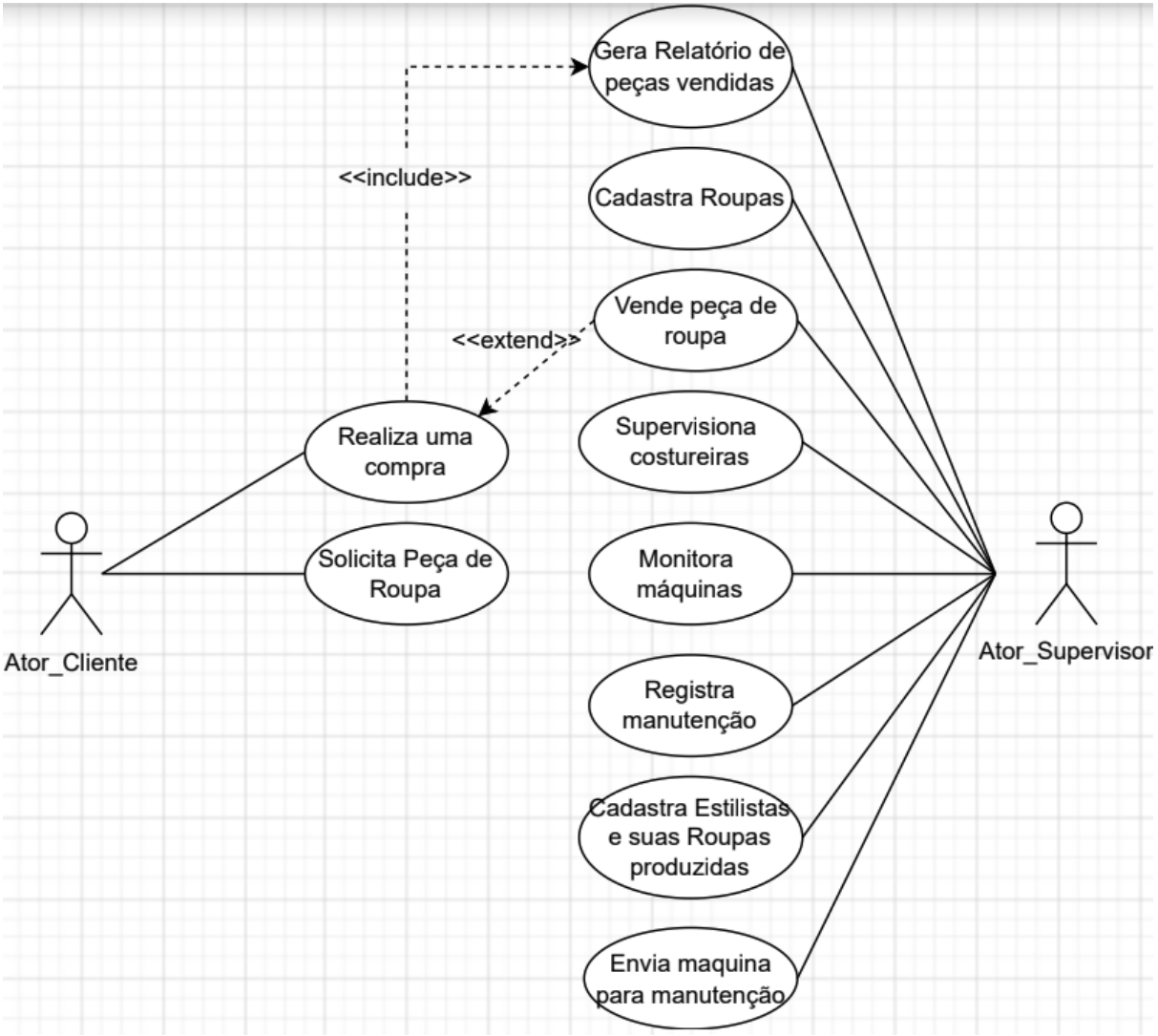
Etapa 1 - Modelagem estrutural e dinâmica em alto nível de abstração

Diagrama de Classes



Foi desenvolvido para esse domínio de modelagem o diagrama de classes acima, ao todo tivemos a necessidade de criar onze classes para que possamos representar nosso sistema da melhor forma possível. Dividindo esse diagrama em dois, temos ali à direita algumas classes que se relacionam com peças de roupa, por exemplo venda, Estilista, TipoDeTecido e estoque, e suas ações respectivamente. Basicamente, podemos ter o processo de um cliente solicitar uma roupa, que por sua vez o estilista vai criar, e irá desencadear toda a cadeia de atividades. Olhando para a esquerda, temos as classes de produção em si, costureiras, supervisores, máquinas, manutenções. Conseguimos cobrir todo o processo de produção de uma peça, de manutenção das mesmas.

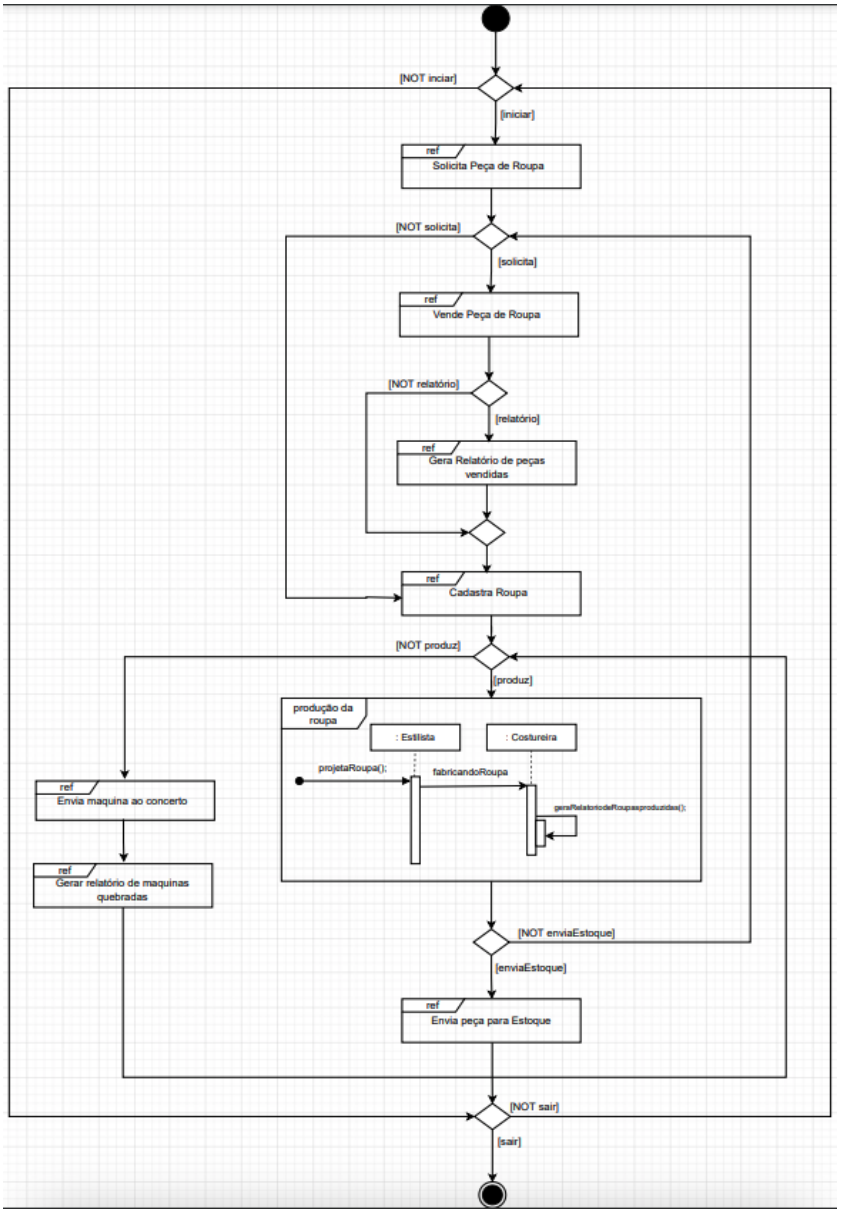
Casos de uso



Acima temos o diagrama de caso de uso que o grupo modelou. Nós separamos os atores em dois: Ator cliente e ator supervisor. O ator supervisor tem bastante poderes e responsabilidades. Eles têm visão sobre as costureira, as roupas que a costureira produz, e as máquinas que estão sendo utilizadas ou estão dentro do estoque. Também tem poderes de pode enviar a máquina para a manutenção e retirá-las de lá. O ator cliente por sua vez, pode ter o ato de solicitar Peça de Roupa, e Realizar uma compra, que por sua vez, está atrelada a função de gerar relatório de peças vendidas do supervisor, e a atividade do supervisor de

Vende peça de roupa, está estendendo para a função de realizar compra do cliente.

Visão geral de interação.



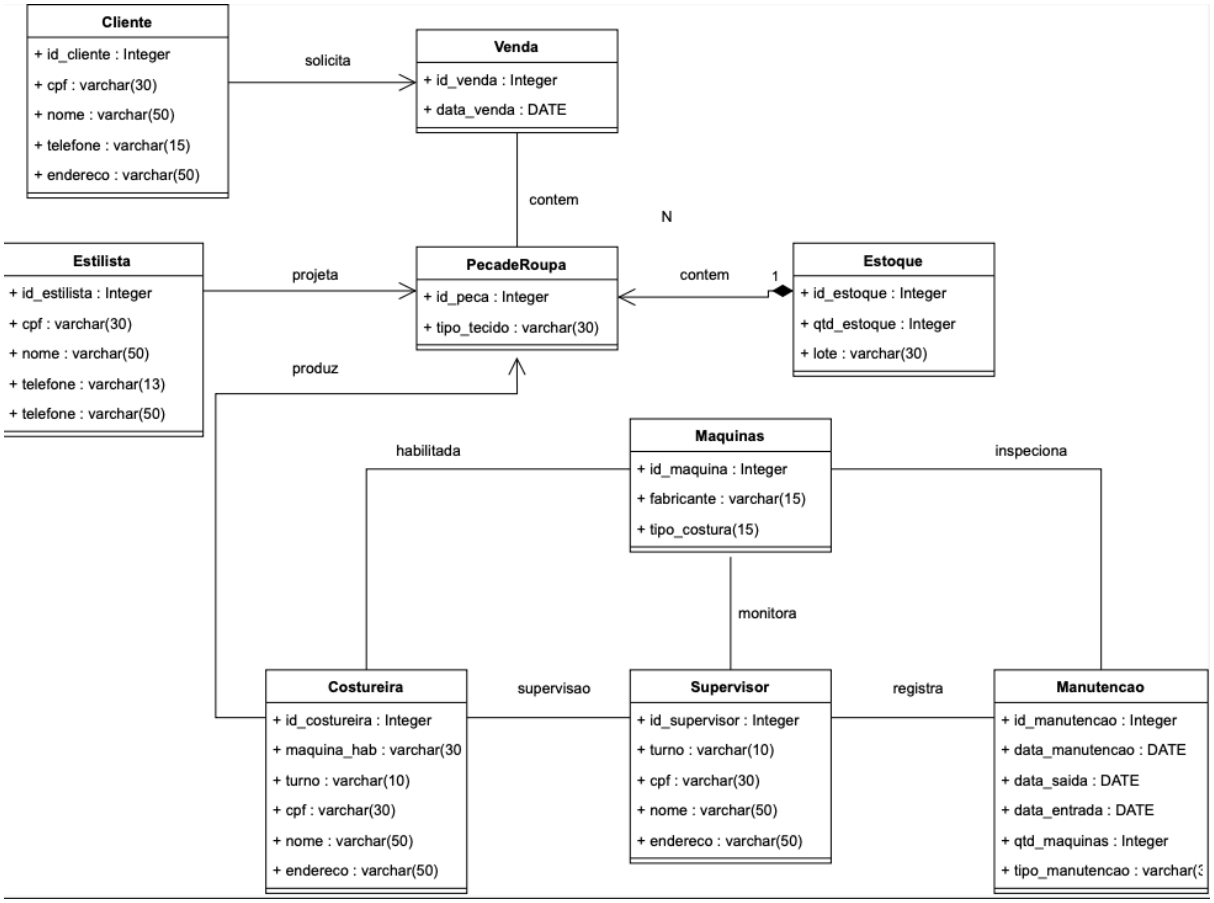
O diagrama de visão geral de interação é um diagrama que mostra interações não necessariamente ligadas a ações, mas sim a outros diagramas feitos anteriormente. Escolhemos como atividade principal para modelarmos esse diagrama o evento de que um cliente solicita uma peça de roupa, até o momento em que essa roupa é entregue ao respectivo cliente. Abordamos possíveis acontecimentos que podem aparecer no caminho, como o fato de uma máquina

pode quebrar. No início dessa modelagem, caso o processo seja iniciar, vai para o nodo de Solicitar Peça de Roupa, caso o cliente solicite, irá para o nodo de vender a peça de roupa, logo após, se for já gerar um relatório desta venda, vai para o nodo de Gerar o relatório de peças vendidas, cado não, vai direto para o de cadastrar Roupas.

Logo após cadastrar essa roupa, entramos no processo propriamente dito de produção, se a peça não estiver em estoque, ela entrará no diagrama de sequência de produção de roupa, que será explicado ao decorrer deste relatório. Caso a roupa seja produzida, e não vá ao estoque, ela já entrará no processo de venda. Nesse processo, é importante ressaltar que caso a máquina quebre no nodo de pergunta se vai produzir ou não, entraremos na tratativa de duas ações, Enviar máquina ao concerto e Gerar relatório de máquinas quebradas. quando essa máquina volta do concerto, a produção nela será retomada. Quando a peça é vendida, ou enviada para o estoque, entraremos no final do processo.

Etapa 2 - Refinamento estrutural

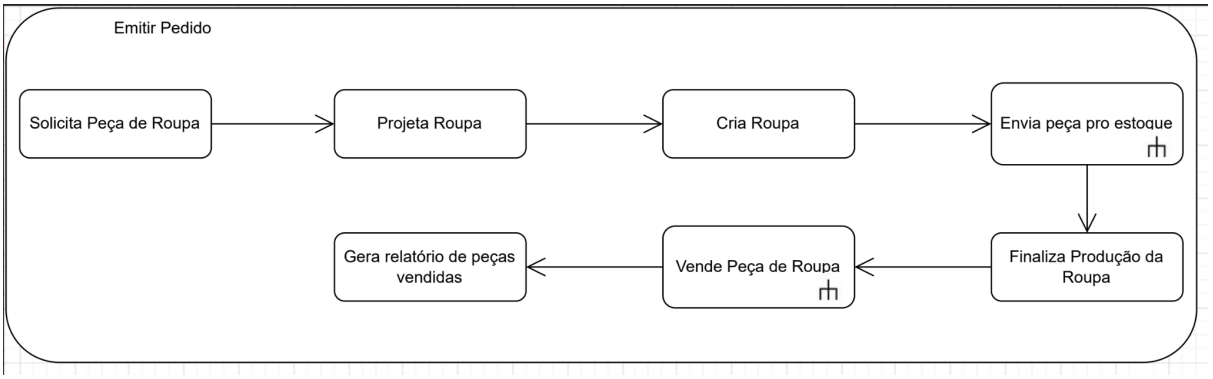
Diagrama de classes refinado



Acima temos o diagrama de classes refinado, é uma melhoria do diagrama de caso de uso, usado para a identificação de atributos, observando as classes e os casos de uso, tem como objetivo identificar componentes de um sistema estão relacionados entre si ,atributos de cada classe e características importantes do problema tratado. Assim, abordamos no diagrama os principais atributos e foi feito todas as conexões do diagrama de uso, como registrar dados importantes (cpf, turno, nome, endereço, entre outros) de cada classe, como as chaves primárias de cada entidade para uma futura aplicação a um banco de dados

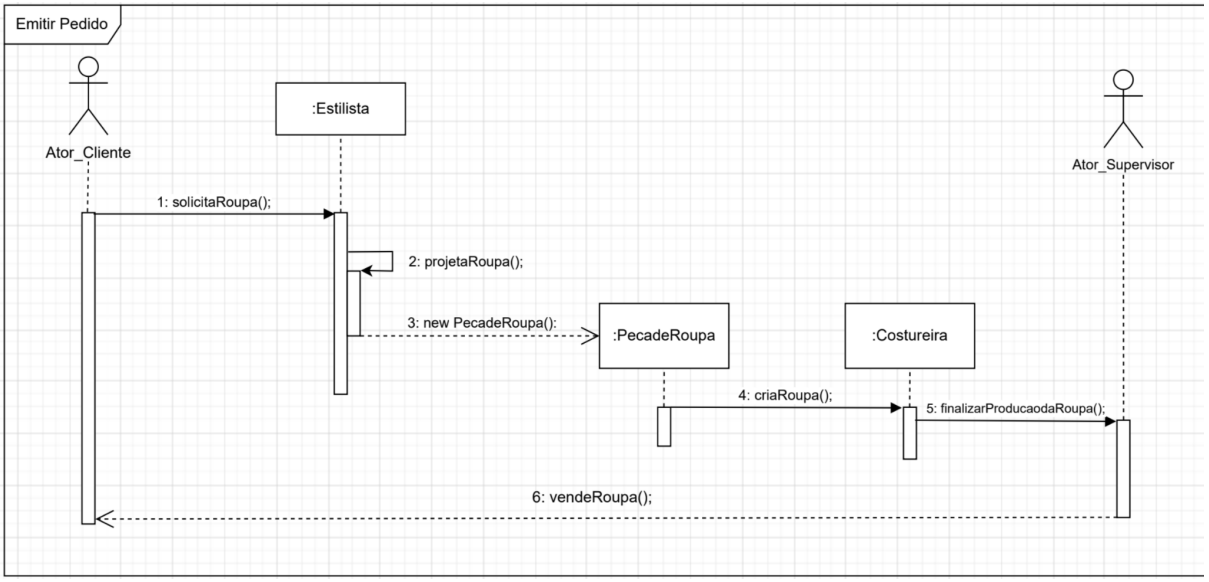
Etapa 3 - Refinamento de casos de uso

Diagrama de atividades (cenário principal)



Acima temos o diagrama de atividades, que é uma melhoria do diagrama de caso de uso, basicamente nesse diagrama, pegamos um processo, e colocamos no diagrama passo a passo para a execução completa deste processo. O escolhido como atividade principal, foi a atividade de emitir pedido. Primeiramente temos a Solicitação de Peça de Roupa, que é feita pelo cliente, passa para o Projeta roupa, que é responsabilidade do estilista, que passa para a costureira criar a roupa, ela envia a peça para o estoque. Esse campo está com o Tridente com referência a ser um processo forçado, pois é preciso que toda peça de roupa produzida antes de ser vendida passe pelo estoque. Em seguida, finalizamos a produção de Roupas seguido da venda da peça de roupa. Ao final desse processo, podemos fazer a Geração de relatório de peças vendidas.

Diagrama de Sequência.



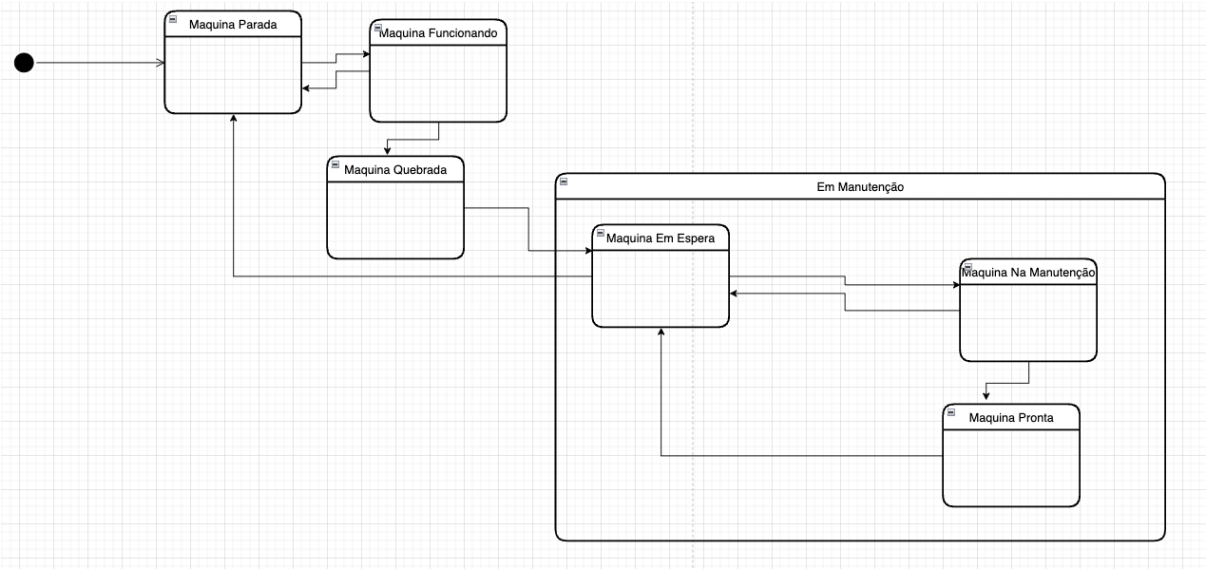
Esse é o diagrama de sequência, que por sua vez, representa a mesma coisa que o diagrama de atividades, mas agora enfatizando a ordem de chamadas de funções , criação de objetos no sistema e o retorno delas .Como por exemplo no estilista, ele projeta a roupa, mas ela volta para ele para que ele as envie para a costureira. Nessa ilustração temos como base a emissão de um pedido e suas etapas sendo feitas sucintamente, o início de toda essa ação começa quando o cliente realiza a solicitação da peça de roupa, passando a compromisso para o estilista que fica responsável por projetar a roupa e em seguida emitir o novo projeto de peça de roupa criado para a costureira produzir e tecer como exigido pelo estilista. Ao final do processo de criação a costureira finaliza e passa a roupa produzida para o supervisor realizar a venda para o cliente.

Em suma, um diagrama de sequência mostra a ordem e a relação temporal entre eventos em um sistema, enquanto um diagrama de atividade mostra as atividades e os fluxos de trabalho em um processo. Um diagrama de sequência é útil para entender como diferentes objetos em um sistema interagem entre si, enquanto um diagrama de atividade é útil para entender como um processo é

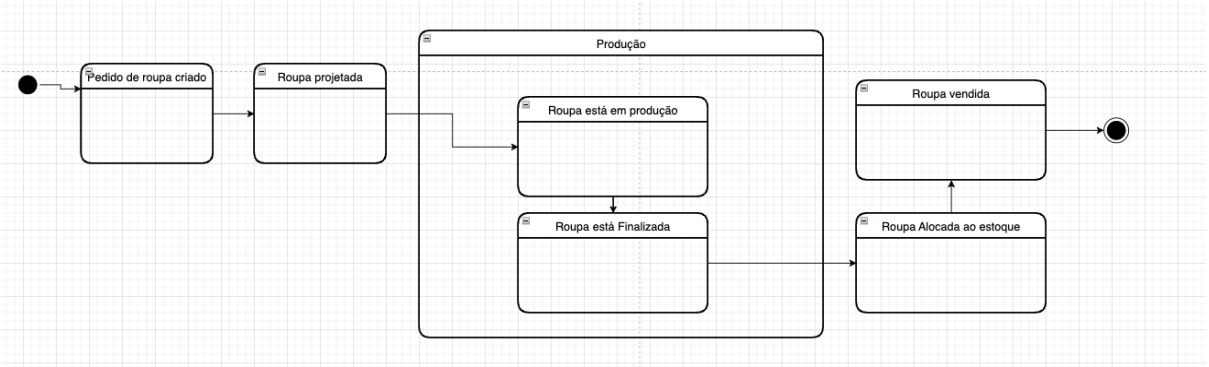
executado e quais tarefas são realizadas. Em resumo, a principal vantagem de utilizar o diagrama de sequência é a de entender a ordem e a relação temporal de eventos de um sistema.

Etapa 4 - Modelagem de estados associada à classe

Diagrama de Máquina de estado

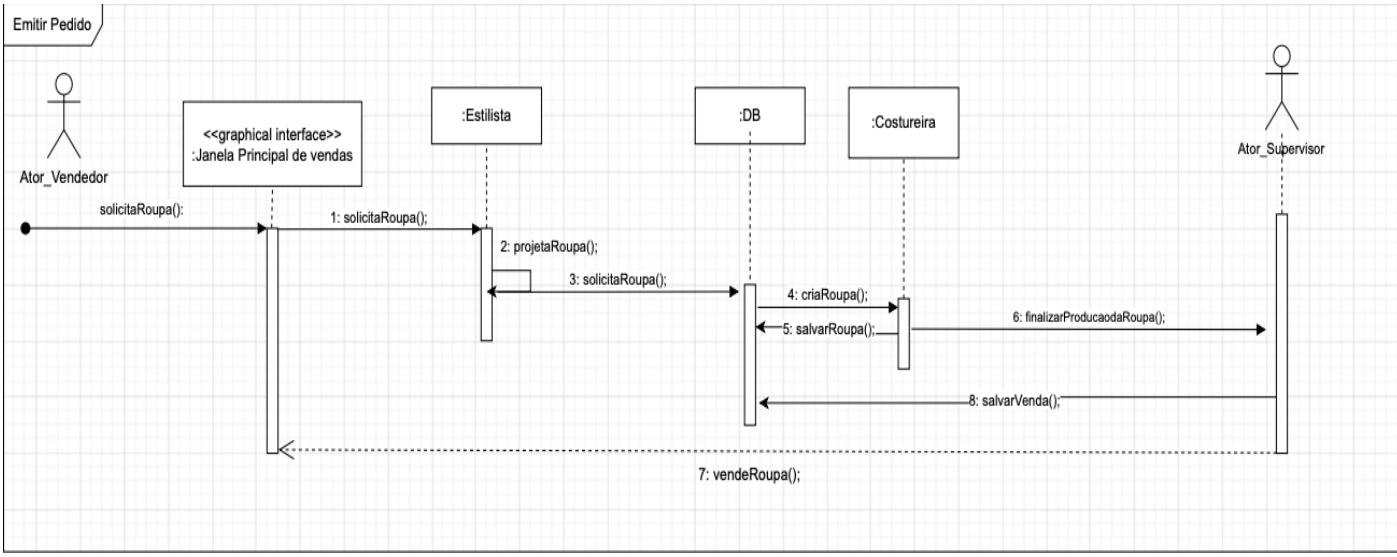


O diagrama de estado basicamente tem o foco de demonstrar o estado em que o sistema ou um objeto se encontra ao decorrer do processo. Acima temos o diagrama de estado de uma máquina de costura. Temos no início a máquina parada, pois ela não está em uso para produção. Logo após, essa máquina pode entrar em uso, que é quando ela está produzindo uma peça de roupa. Ela pode voltar para o estado de máquina parada caso tudo ocorra bem, ou ela pode entrar no estado de quebrada. Caso ela entre no estado de parada,desloca para o subdiagrama de manutenção, que dentro contém os estados de máquina em espera, que é quando a máquina chega na manutenção e entra na fila de espera para a manutenção.Em seguida, ela entra em no estado de máquina em manutenção e pode voltar para espera caso ocorra algum problema, ou ela vai para pronto, e consequentemente volta para o estado de espera para poder sair da manutenção. Saindo da manutenção a máquina volta para o estado de parada e reinicia todo o ciclo.



Acima temos o diagrama de estado para as roupas, a roupa entra no estado de estar sendo produzida, passa para o estado de produção iniciando em já em produção, quando ela já está finalizada, e logo ela entra no estado para poder entrar no estoque, entrando no estoque essa peça é vendida e o objeto sendo destruído.

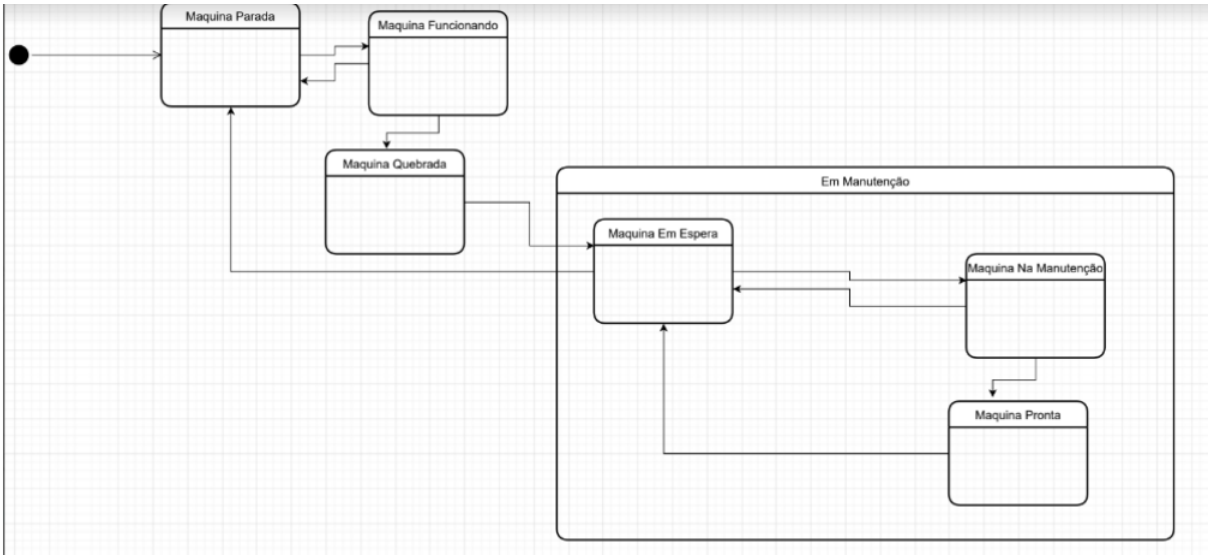
Etapa 5 - Introdução de elementos do domínio da solução computacional.



Acima temos um diagrama de elementos do domínio da solução computacional. Basicamente implementamos neste diagrama, a representação de como seria o fluxo desse processo quando inserimos questões tecnológicas ao

sistema, como a interface gráfica, e questões de armazenamento, de como seria persistido as informações. Podemos destacar também questões de quem vai usar o sistema. Frisamos ao decorrer da projeção que temos dois atores que vão usar o sistema, supervisor e cliente, mas quando olhamos essas questões técnicas, percebemos que o cliente vai ser abstraído pelo vendedor, que de fato irá gerir os processos do cliente.

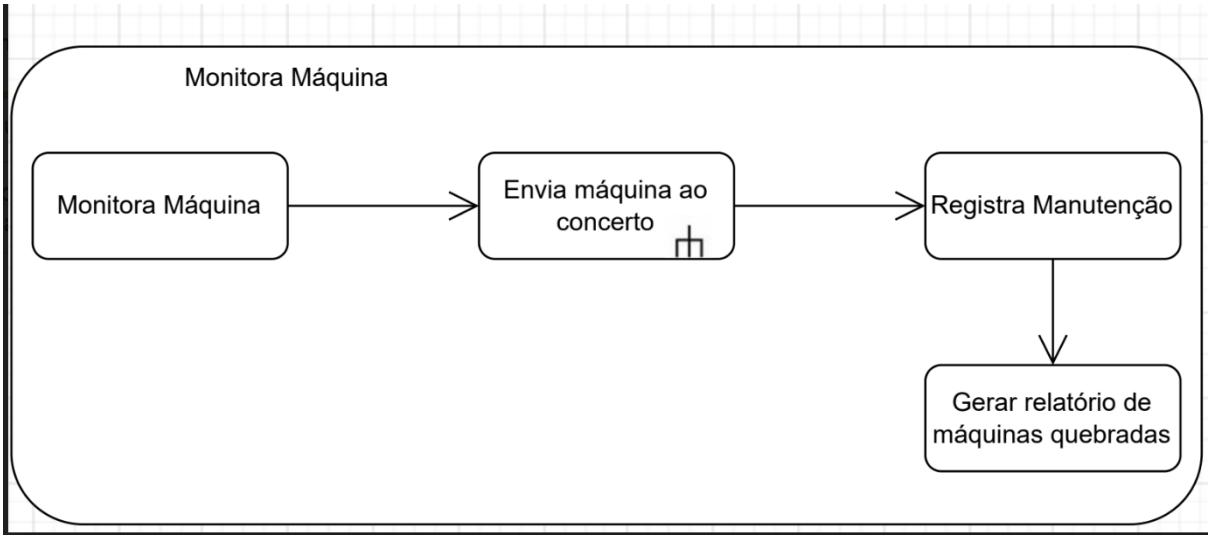
Etapa 6 - Destaque situações especiais



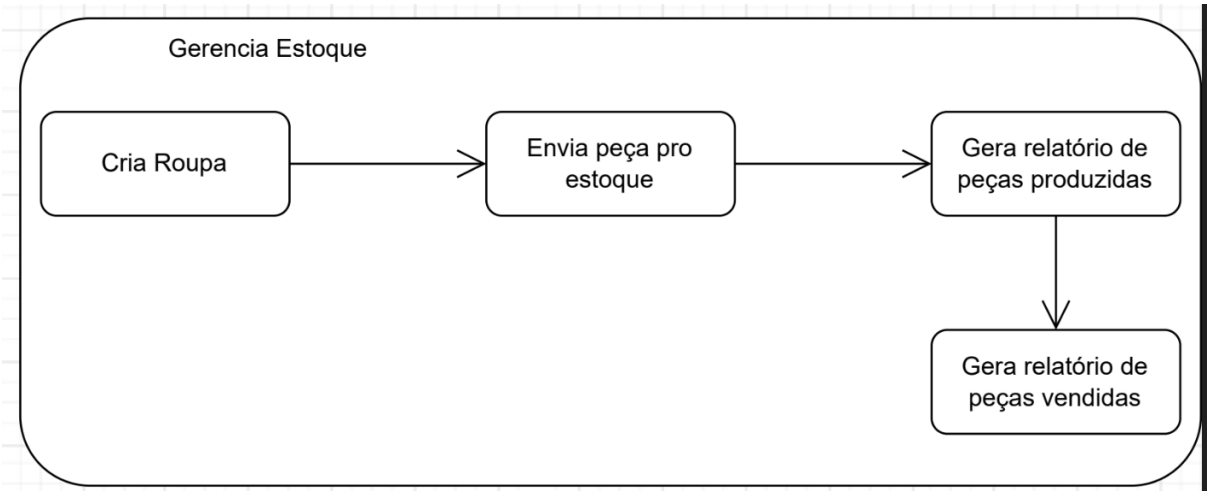
Basicamente decidimos dar ênfase à fase de análise das condições das máquinas e suas fases dentro do sistema, retratada na etapa 4 de modelagem.

Etapa 7 - Modelagem de algoritmos de métodos

Diagrama de Atividades



Acima temos o diagrama de atividades que está representando o processo de monitoração de máquinas que envolve as atividades de enviar máquinas para o concerto e removê-las de lá. No início dessa atividade temos o processo de Monitorar Máquina, que é feio por um supervisor, que vai enviar essa máquina para o concerto, essa atividade por sua vez tem o atributo de ser forçado, que na verdade é que de alguma forma nesse processo, tem que passar por aquela atividade. Logo após temos a de registrar manutenção, que é feita pela classe de manutenção. E caso o supervisor precise, ele consegue gerar um relatório futuro com as máquinas que estão quebradas.



Acima temos o diagrama de enviar peça de roupa ao estoque. Esse diagrama por sua vez, demonstra o passo a passo das atividades até termos a possibilidade de gerar um relatório das máquinas que estão no concerto. Começamos pelo

processo de criar roupa, que é outro diagrama de atividades, e passamos para o de enviar peça pro estoque. Após isso podemos produzir relatórios de peças produzidas e de Geração de peças vendidas.

Etapa 8 - Geração de código e o desenvolvimento iterativo

Logo após a geração de todos os modelos e diagramas, chegou a hora de colocarmos a mão na massa. Para isso, começou a fazer a criação das classes existentes em todo o processo de modelagem.

Com os diagramas em mãos, fica extremamente mais fácil de fazer a montagem do código, pois a parte mais complicada já foi pensada previamente.

Mostrando primeiramente a classe de cliente temos a seguinte configuração:

Cliente:

A anotação @Data e @AllArgsConstructar é de uma biblioteca específica que esconde os métodos get's e set's e não vem ao caso. Podemos observar nessa classe os atributos criados e seu método principal.

Costureira :

```java
@Data
@AllArgsConstructor
public class Costureira {
    private String id;
    private String nome;
    private Roupa roupaAtual;
    private MaquinaDeCostura maquinaDeCosturaAtual;

    private List<String> tipoDeMaquinaPossuiHabilitacao = new ArrayList<>();
    private String turno;
    private Supervisor supervisor;
    private List<Roupa> roupas = new ArrayList<>();

    Guilherme Dias Cardoso
    public void finalizarProducaoDaRoupa(Roupa roupa, Estoque estoque) {
        System.out.println("Finalizando a produção da roupa e enviando ao estoque");
        estoque.addRoupa(roupa);
        this.setRoupaAtual(null);
    }

    Guilherme Dias Cardoso
    public Roupa criaRoupa(Roupa roupa) {
        System.out.println("Costureira produzindo a roupa ");
        roupa.setCostureira(this);
        roupa.setTipoDeTecido(maquinaDeCosturaAtual.getTipoDeTecido().get(0));
        roupas.add(roupa);
        this.setRoupaAtual(roupa);
        this.getMaquinaDeCosturaAtual().setRoupaAtual(roupa);
        return roupa;
    }

    Guilherme Dias Cardoso
    public void gerarRelatorioDeRoupasProduzidas() {   Cardoso, 09/01/23 22:56 • .
        for (Roupa roupa : roupas) {
            System.out.println("=======Gerando relatório de ropas produzidas por COstureira");
            System.out.println("Roupa produzida: " + roupa.getId());
            System.out.println("Tipo de tecido: " + roupa.getTipoDeTecido().getNome());
            System.out.println("Produzida por: " + roupa.getEstilista().getNome());
            System.out.println("Fabricada por: " + roupa.getCostureira().getNome());
        }
    }
```

Supervisor:

```java
@Data
@AllArgsConstructor
public class Supervisor {
    private String id;
    private String turno;
    private String cpf;
    private String nome;
    private String edereco;
    private List<Costureira> costureiras;
    private List<MaquinaDeCostura> maquinaDeCosturaHashMap = new ArrayList<>();

    Guilherme Dias Cardoso
    public void gerarRelatorioDeMaquinasQuebradas() {
        System.out.println("=========Gerando relatório de maquinas quebradas");
        for (MaquinaDeCostura maquinaDeCostura : maquinaDeCosturaHashMap) {
            if (maquinaDeCostura.isEstaEmManutencao()) {
                System.out.println("A maquina com o Id: " + maquinaDeCostura.getId() + " esta em manutenção!");
            }
        }
    }

    Guilherme Dias Cardoso
    public Manutencao enviarMaquinaAoConcerto(MaquinaDeCostura maquina) {
        System.out.println("Enviando a maquina: " + maquina.getId() + " ao concerto!");
        addMaquina(maquina);
        return new Manutencao(maquina);
    }

    Guilherme Dias Cardoso
    public Venda venderPecaAoCliente(Estoque estoque, Roupa roupa, Venda venda) {
        System.out.println("Realizando a venda ao cliente!");
        estoque.getRoupas().remove(roupa);
        venda.addRoupas(roupa);
        return venda;
    }   Cardoso, 09/01/23 22:56 • .
    Guilherme Dias Cardoso
    public void removeDaManutencao(Manutencao manutencao, MaquinaDeCostura maquinaDeCostura) {
        System.out.println("Removendo a maquina: " + maquinaDeCostura.getId() + " da manutenção!");
        manutencao.setData_saida(new Date());
        maquinaDeCostura.setEstaEmManutencao(false);
    }
```

Bom, essas foram as classes principais e mais importantes do modelo, abaixo temos a função main, com dada diagrama de atividade claramente implementados, explicando primeiramente a função emetirPedido:

```java
Guilherme Dias Cardoso
public static void emitirPedido() {
    Cliente cliente = new Cliente( id: 1,  cpf: "123456789",  nome: "João",  endereco: "Rua 1",  telefone: "123456789");
    Estilista estilista = new Estilista( id: 1,  cpf: "123456789",  telefone: "35984114444",  endereco: "Rua 1",  nome: "Joao");
    Supervisor supervisor = new Supervisor( id: "1",  turno: "Manha",  cpf: "123456789",  nome: "João",  edereco: "Rua 1");
    MaquinaDeCostura maquinaDeCostura = new MaquinaDeCostura( id: 1, List.of(new TipoDeTecido( id: "1",  nome: "Algodão")), List.of(new TipoDeCostura( id: "3",  nome: "Ponto zig-zag")),  supervisor: null,  fabricante: "Embraer",
    Costureira costureira = new Costureira( id: "1",  nome: "Marcia",  roupaAtual: null, maquinaDeCostura,  turno: "Manhã", supervisor);
    Estoque estoque = new Estoque( id: "0",  qtde: 0, new ArrayList<>());

    System.out.println("Processo de emitir Pedido");
    Venda venda = cliente.solicitarRoupa( roupa: "Vestido Vermelho",  quantidade: 2);
    Roupa roupa = estilista.projetaRoupa( nome: "Vestido Vermelho");
    Roupa roupaProduzida = costureira.criaRoupa(roupa);
    costureira.finalizarProducaoDaRoupa(roupaProduzida, estoque);
    Venda vendaFinal = supervisor.venderPecaAoCliente(estoque, roupaProduzida, venda);
}
```

Primeiramente, dentro dessa função conseguimos ver a declaração de atributos fictícios que são necessários para a execução prévia do programa, logo após chamamos um atributo do cliente, que é solicitar Roupa passando a roupa e a quantidade. Isso podemos fazer com que o estilista projete a roupa passando o nome da roupa, e o estilista retorna a roupa, com essa roupa passamos para a costureira fazer a roupa e finalizá la com o estoque, porque dentro dessa função ela já manda para o estoque

Outro processo do fluxo de atividades é a função de criar Roupa:

Essa função também precisamos previamente fazer a manipulação de alguns arquivos fictícios para a clara execução, começamos projetando a roupa como na execução da função acima, e logo após a costureira cria a roupa recebendo o objeto da roupa e devolve a roupa já enriquecida com a informações necessárias, por fim a costureira utiliza a função de finalizar a produção dessa roupa e por fim fazer a geração do relatório de roupas produzidas até o momento. Esse último método é opcional.

O método de monitorar estoque é mais simple, tendo poucos objetos necessários para ser criados:

Precisamos apenas de um supervisor e de uma máquina de costura para que esses procedimentos possam acontecer. Com isso, chamamos o método do supervisor que envia a máquina ao concerto, passando a máquina, com isso, o supervisor vai colocá-lo em uma lista interna deles de máquinas no concerto, e

criará um atributo de manutenção e colocará a máquina dentro. Logo após, chamamos o método da manutenção, que é registrar a manutenção que vai de fato alterar todos os atributos da máquina para demonstrar que ela está no concerto. Após isso, por opção fazemos uma geração de um relatório. Por final, o supervisor tem um método que remove a máquina da manutenção, e altera todos os atributos necessários para a máquina voltar ao funcionamento normal dela.

Por fim, temos a função de **gerenciarEstoque**:

Para o funcionamento desse método, precisamos da criação prévia de alguns atributos: estilista, supervisor,máquina De Costura,costureira,estoque.

Começamos com o estilista projetando a roupa sendo passado o nome da roupa, logo após jogamos esse objeto da roupa que foi gerado dentro do método da costureira cria Roupa, que vai enriquecer o atributo da roupa, e gerar um roupa produzida. Após isso nós podemos finalizar a produção da roupa passando a roupa que foi produzida e o estoque para que a costureira envie para o estoque a roupa. com isso, podemos gerar relatórios das roupas vendidas e de roupas produzidas.