2 Conceitos Básicos

Este capítulo apresenta as tecnologias envolvidas na solução do problema proposto e uma revisão bibliográfica sobre o assunto abordado na dissertação. O capítulo aborda inicialmente a modelagem de planos, incluindo o tratamento de falhas e exceções. Prossegue abordando as técnicas e teorias utilizadas na solução dos problemas de escalonamento integrado com o planejamento, considerando restrições temporais e o tratamento das exceções e erros que se apresentam em tempo de execução.

2.1 *Workflow*s, Planos e Flexibilidade

Um *workflow* pode ser definido como um conjunto de atividades com uma possível especificação do fluxo de controle e do fluxo de dados entre as atividades. Uma atividade pode ser executada por um ou mais sistemas de computação, por um ou mais agentes humanos, ou então por uma combinação destes.

Um *plano* é um caso particular de um workflow em que se considera apenas composição sequencial e paralela de atividades.

O suporte necessário para flexibilização de *workflow*s baseados no *Workflow Mamagement System* (WFMS) deve ser capaz de lidar com dois desafios fundamentais. O primeiro, a *flexibilidade a priori*, cobre a especificação das atividades a serem executadas com maior precisão, permitindo um comportamento menos restritivo, obtendo alguma flexibilidade, controle adaptável e mecanismos de fluxo de dados que levam em conta as medidas que dão suporte a trabalhos *ad hoc* e cooperativos ao nível do *workflow*.

O segundo desafio, a *flexibilidade a posteriori*, ou o que chamamos de *adaptação dinâmica*, refere-se à alteração e evolução das especificações do modelo a nível de instância e de esquema, tendo como resultado final alterações

no processo real. No caso de modificações dinâmicas, deve ser especificado o contexto e em qual estado de execução as modificações serão permitidas, garantindo a consistência semântica e a dinâmica do processo.

2.2 Falhas, Exceções e Restrições

2.2.1 Falhas

Na especificação de um *workflow* normalmente temos que lidar com incertezas, já que nem sempre todas as informações necessárias estão disponíveis em tempo de especificação. A complexidade do domínio ou as dificuldades que surgem de tempos em tempos levantam à necessidade de ser utilizada uma abordagem mais flexível, admitindo atividades que não podem ser definidas antes da execução.

A execução de um *workflow* está sujeita aos seguintes tipos de falhas:

1) Falhas no executor do *workflow* relacionadas ao término anormal da execução do *workflow*.

2) Falhas em atividades primárias do sistema de *workflow*.

3) Falhas de comunicação entre o escalonador e o executor, que podem levar à não execução de determinada atividade.

Podemos ainda classificar as falhas em *básicas* ou *da aplicação*. As falhas do primeiro tipo são aquelas relacionadas com o sistema responsável pelo gerenciamento de *planos*, por exemplo, quando uma aplicação é chamada e o ambiente aonde será executada a tarefa não responde ou esta indisponível. Já as falhas da aplicação são relevantes pois resultam em falhas em uma ou mais atividades que estão sendo executadas, gerando como conseqüência uma atividade no contexto que não é efetivamente completada. Estes fatos geram a necessidade de ferramentas ou instrumentos que permitam tratar falhas em tempo de execução. O gerenciamento das modificações necessárias pelas aplicações para a sua execução correta normalmente encontra-se fora do domínio do *workflow*.

Em [Blythe, 2003] é utilizada a abordagem transacional para lidar com falhas, que considera a integração de módulos do *workflow* como modelos transacionais. Este trabalho discute a capacidade de realizar o *rollback* parcial ou total do processo até que um ponto estável seja encontrado, com objetivo de corrigir o problema encontrado.

2.2.2 Exceções

Exceções também podem ser chamadas de *falhas semânticas* e ocorrem quando atividades não podem ser executadas como planejado, ou então quando não atingem o resultado esperado. A ocorrência de uma exceção pode ser resultado de inconsistências nos dados, divergência de atividades em relação ao modelo original, contingências inesperadas ou mudanças não modeladas no ambiente como fruto das simplificações e aproximações realizadas durante a fase de modelagem.

Em [Weld, 1994], as exceções são descritas como termos de um conjunto completo de regras disponíveis. A abordagem apresentada em [Penberthy, 1994] utiliza um conceito similar, indicando que exceções são "casos em que o sistema de computação não processa corretamente uma tarefa sem intervenção manual". [Blythe, 2003] considera ainda desvios esperados do fluxo normal como exceções.

A maneira usual de lidar com exceções consiste em tratá-las formalmente na definição do *workflow* ou em tempo de execução. Na primeira abordagem, deve-se considerar quais são as possíveis falhas ou exceções que podem ocorrer e, a partir daí, definir medidas a serem tomadas caso ocorram falhas ou exceções após cada atividade. Como desvantagem, esta estratégia normalmente requer considerar um número muito grande de possibilidades, geralmente de ordem exponencial, o que torna o *workflow* difícil de ser gerenciado e de ser compreendido.

O tipo da exceção tem impacto direto no tipo de solução e no tratamento a ser adotado, nas mudanças que serão necessárias no modelo do *workflow* para

corrigí-las e também no tipo de suporte a ser fornecido. As exceções podem ser classificadas de duas formas. A primeira considera os seguintes tipos de exceções:

1) *Exceções esperadas* são desvios previsíveis do comportamento normal do processo. Este tipo de exceção está estritamente relacionado com o domínio do *workflow*, sendo parte da semântica do processo, e pode ser modelada com antecedência. Este tipo de exceção pode ser subdividida em 4 subtipos, de acordo com os eventos que as originam:

   a. *Exceções do workflow* surgem quando do início ou término das atividades e, portanto, estão sincronizadas com o progresso do *workflow*.

   b. *Exceções dos dados* surgem das modificações em dados relevantes do *workflow*.

   c. *Exceções temporais* são ocasionadas pela ocorrência de determinado *timestamp*, periodicidade ou como intervalo definido com referência a determinado evento.

   d. *Exceções externas* são causadas por eventos externos, notificados de forma explícita para o motor de *workflow* por agentes ou aplicações externas.

      Geralmente quando ocorre este tipo de exceção, é necessário desfazer algumas tarefas anteriores (como forma de compensação), até chegar num estado consistente e, então, tentar retornar à execução por um caminho alternativo, buscando chegar ao final do processo.

2) *Exceções Inesperadas* lidam com a necessidade de mudança da estrutura do processo corrente do *workflow*, normalmente em tempo de execução, como resultado de uma exigência totalmente nova, ou desvio imprevisto do fluxo do processo.

A segunda forma considera os seguintes tipos de exceções:

1) *Exceções de ruído* são exceções toleradas pelo processo ou que podem ser ignoradas com segurança e, mesmo que ocorram, produzem resultados satisfatórios.

2) *Exceções idiossincráticas* são relativamente únicas e especificas a instâncias do trabalho e necessitam que sejam feitas mudanças nos processos destas instâncias para que possam ser corrigidas.

3) *Exceções evolucionárias* necessitam de mudanças no modelo do *workflow* como um todo e podem levar ao replanejamento ou até a uma nova modelagem do processo.

2.2.3 Restrições

Esta seção lista as principais categorias de restrições, conforme [Fox, 1985]:

1) *Restrições organizacionais*,.
   a. *Restrições devidas a datas*. Referem-se a limites estabelecidos de datas, que podem afetar a entrega do produto final.
   b. *Restrições sobre o nível de recursos*. Referem-se à existência de recursos necessários para manter as operações, onde cada recurso pode estar associado a uma restrição.
   c. *Restrições sobre custos*. Estão relacionadas com o tempo final para a realização de uma tarefa, alocação de determinado equipamento ou ate mesmo de pessoal.
   d. *Restrições sobre o nível de produção*. Permitem estimar as metas de produção de cada centro de custos (máquina/equipamento), auxiliando na tarefa de distribuição de trabalhos.
   e. *Restrições sobre estabilidade*. Estão ligadas à qualidade do escalonamento e da quantidade de revisões necessárias.
   f. *Restrições sobre o tempo de funcionamento*. Estão intimamente relacionadas com a disponibilidade dos recursos e qualidade do serviço.

g. *Restrições sobre qualidade*. Estão relacionadas com o serviço oferecido e com a confiabilidade.

h. *Restrições sobre metas de produção*. Estão relacionadas com problemas e deficiências no cumprimento de prazos e realização de tarefas.

2) *Restrições físicas*. Estão relacionadas com as características físicas dos equipamentos, com a capacidade de processamento, com o espaço disponível em disco, etc.

   a. *Restrições sobre equipamentos*. São as restrições características de cada equipamento que influenciam diretamente no processo.

   b. *Restrições sobre o funcionamento de equipamentos*. Estão relacionadas com a taxa de disponibilidade e confiabilidade dos equipamentos.

   c. *Restrições sobre qualidade*. São restrições relacionadas a qualidade do produto final da execução das tarefas, estando relacionadas com as características dos equipamentos disponíveis.

3) *Restrições causais*. Estão relacionadas com a satisfação de determinadas condições antes da execução das operações.

   a. *Restrições sobre precedência*. São restrições relacionadas com a seqüência de eventos quando uma ação, relaciona-se com por exemplo

   b. *Restrições sobre a necessidade de recursos*.

4) *Restrições sobre disponibilidade de Recursos*

   a. *Restrições sobre tempo parado*

   b. *Restrições sobre reserva de recursos*

   c. *Restrições sobre deslocamento*

5) *Restrições sobre preferência*

   a. *Restrições sobre operação*

   b. Restrições sobre equipamento

   c. *Restrições sobre preferência de fila*

Podemos ainda classificar as restrições como:

1) *Restrições de recursos*. Descrevem as limitações relacionadas ao tempo e a utilização dos recursos (capacidade, compatibilidade e etc).
2) *Restrições de transição*. Descrevem o comportamento de um único recurso no tempo.
3) *Restrições de dependência*. Especificam as restrições entre os recursos, normalmente entre fornecedores e consumidores.

Dependendo do modelo escolhido para descrever o problema, as restrições agrupadas como dinâmicas serão introduzidas durante o processo de escalonamento, sendo identificadas três modelos distintos.

1. *Modelo de linha do tempo* (*Time model Line).* Utiliza intervalos discretos de tempo, chamados de fatias ou intervalos de tempo, e considera o comportamento dentro de determinado intervalo de tempo como sendo homogêneo. Assim, por exemplo, uma atividade só irá sofrer qualquer tipo de alteração nas fronteiras entre os intervalos de tempo considerados. Portanto, a duração do intervalo de tempo é computado. O comportamento dos recursos disponíveis é descrito pelo conjunto de valores atribuídos às variáveis em cada intervalo de tempo. Então, as restrições de recursos formam um conjunto para um único intervalo, as restrições de transição vinculam as variáveis de intervalos consecutivos e as dependências de recursos vinculam os recursos de diferentes intervalos.
2. *Modelo centrado em tarefas.* É baseado em atividades grupadas em tarefas. Uma tarefa é definida como uma seqüência de atividades conectadas por restrições de dependência. Usualmente são consideradas somente as restrições mais importantes para o processo. Neste tipo de modelo, os recursos e as restrições de transição são inseridos dinamicamente devido à dificuldade de representar os gatilhos estaticamente. Este modelo é o mais comum nos problemas de escalonamento tradicionais por não incluir todas as restrições de transição e por considerar restrições recursos simples. Como, em ambientes e processos complexos, tanto os recursos quanto as restrições de transição têm uma função importante, este modelo torna-se menos adequado.
3. *Modelo baseado em recursos.* É ortogonal ao modelo anterior pois agrupa as atividades por recursos. Este tipo de agrupamento simplifica a forma de

expressar as transições de restrições e de recursos. Porém, em contrapartida, as dependências devem ser inseridas dinamicamente.

2.3 Propriedades e características de sistemas de *workflow* flexíveis

Para que um sistema de *workflow* seja flexível, deve possuir algumas propriedades e características [Eder, 1995]:

1) *Detecção de exceções*. O sistema de *workflow* deve suportar a descoberta de exceções e falhas em tempo de execução para que possam ser tratadas e não causem a interrupção do processo. Os sistemas devem ter pontos de verificação ou regras que permitam verificar o progresso do *workflow*, indicando claramente as possíveis fontes de exceções e falhas. A utilização de agentes reflexivos pode fornecer informações e *feedback* sobre o progresso da execução.

2) *Evitar exceções*. Um *workflow* integrado ao ambiente de trabalho por si só já é causa de condição de exceções. Sistemas rígidos na sua utilização, ou com poucas ferramentas e opções, levam os usuários a operar fora do contexto do sistema. Os sistemas abertos, que podem ser incrementados com abordagens para execução flexível, tendem a evitar conflitos que gerem exceções. Agentes adaptáveis são úteis para testar a efetividade dos componentes do *workflow*.

3) Capacidade de lidar com exceções

   a. *Tolerância a pequenos desvios, ou ruídos*. O modelo de execução deve oferecer alguma tolerância a pequenos desvios.

   b. *Mudanças em instâncias do processo.* São causadas por exceções idiossincráticas como, por exemplo, a seleção de determinado serviço que se encontra indisponível e necessita substituição em tempo de execução. Como solução, o modelo deve ser especificado com uma ordem precisa de execução das atividades, fornecendo restrições e apresentando ao usuário as múltiplas tarefas que possam ser

realizadas, em caso de ocorrerem exceções. Uma abordagem alternativa seria não especificar o modelo com uma ordem precisa de atividades, mas fornecer restrições e apresentar ao usuário múltiplas opções para realizar as tarefas, como uma relação de possíveis tarefas que podem ser executadas, em tempo de execução.

4) *Otimização e evolução do modelo do processo***.** Deve ocorrer durante a evolução dos objetos do processo, como resultado de tarefas, definição de prioridades, responsabilidades e outros fatores. Como resultado de uma otimização, pode ser necessário adicionar, remover ou redefinir atividades ou restrições ao modelo. A arquitetura ou estrutura prevista deve ser capaz de permitir mudanças antes ou depois do alinhamento das tarefas. Algumas otimizações podem ser realizadas pelo próprio sistema através de agentes, ferramentas de validação ou otimizadores; outras otimizações só com intervenção direta do usuário.

Também em [WFMC-TC-1012, 1996], os autores sugerem funcionalidades básicas que dão flexibilidade ao sistema de *workflow* e permitem o gerenciamento de exceções, em tempo de execução:

1) *Alteração e composição dinâmica*. Refere-se à capacidade de modificar dinamicamente as definições de processos, dados e comportamentos associados com objetos que sejam cruciais para a execução do processo.

2) *Composição em tempo de execução.* Em muitos casos, o *workflow* não pode ser completamente especificado antes do início da execução devido a indeterminismo. Este procedimento pode ser realizado através de verificações programadas, que podem ser em dterminados pontos ou momentos pré-determinados e pontos de verificação durante a execução do processo.

3) *Modelos de execução configuráveis*. Refere-se à utilização de um mecanismo que permita reiniciar a execução a partir de determinado ponto onde tenha ocorrido uma interrupção, ou realizar um *rollback*, seja por meio manual ou automático, buscando um replanejamento das ações, em função das novas condições do ambiente.

4) *Execução parcial.* Com a composição dinâmica é possível a execução de fragmentos de processos incompletos. Assim a execução do processo pode ser suspensa e continuar de qualquer ponto específico, incluindo a reorganização dinâmica das relações locais e restrições para o novo contexto.

5) *Orientação e Obrigação.* O sistema de *workflow* deve oferecer mecanismos que permitam variações e controles apropriados através do processo.

6) *Modelagem tipada de atividades, recursos, agentes e artefatos*. Permite que não somente informações sobre componentes estejam disponíveis, mais também que classes equivalentes de objetos possam ser determinadas. A modelagem tipada permite que sejam feitas escolhas em tempo de execução com garantia de que o impacto seja mínimo sobre a consistência e desempenho como um todo.

7) *Reflexividade.* Permite que um processo, ao ser executado, tenha a possibilidade de se auto-remodelar, seja automaticamente ou através de intervenção externa, em resposta a um feedback de como o processo está se comportando. Baseado em informações sobre o ambiente, um componente deve ser capaz de interagir com novos elementos do ambiente, otimizar o escalonamento e as restrições, e melhorar a organização das relações e dependências. A utilização de regras ou estratégias especificas podem direcionar o *workflow* reflexivo, buscando ser capaz de aumentar a eficiência de forma eficaz, em resposta as medidas qualitativas coletadas.

8) *Regras e estratégias específicas.* Refere-se à capacidade do *workflow* se auto-desenvolver, com o objetivo de permitir otimizações localizadas ou generalizadas.

9) *Decomposição do processo em modelos lógicos.* Através da fragmentação lógica do processo é possível evitar, conter e recuperar exceções. A utilização de abstrações facilita a modelagem e aumenta a compreensão, especificando pontos de interação entre os sub-processos, permitindo o ajuste e lidando com exceções, em tempo de execução. O processo pode

mudar baseado no *feedback* do usuário, na utilização, no nível de iteração e no número de vezes que foi executado.

10) *Desenvolvimento de modelos de processos*. Refere-se à definição de modelos genéricos de *workflows* que podem servir como guia para outros *workflows* .

11) *Bibliotecas e fragmentos de processos reutilizáveis*. Fragmentos são seqüências de atividades que, associadas a determinados recursos, podem fornecer padronização para tarefas complexas. Através destes mecanismos, podemos prover melhor suporte ao processo de evolução e otimização. Assim, fragmentos devem ser facilmente reutilizáveis, divisíveis e capazes de serem avaliados contra alguns critérios.

12) *Arquitetura para monitoramento de eventos*. Exceções podem se manifestar através de ocorrências simples ou inconsistências no ambiente de trabalho e não serem detectadas. Para lidar com este tipo de ocorrência, a utilização de componentes reflexivos e o armazenamento de objetos, que através de mudanças detectadas nos dados do processo, ativa rotinas que fornecem avisos quando alguma restrição é violada, alguma exceção é detectada ou o limite de tempo é excedido.

Em outra direção, identificamos três tipos de modificações que podem ser utilizadas durante o processo de reparo de um plano:

1) *Re-expandir*. Ocorre quando assumimos que determinada condição falhou. A modificação consiste em adicionar ou remover um sub-plano. Esta operação é uma opção quando não queremos afetar o plano como um todo resolvendo somente o problema local.

2) *Reajustar.* Ocorre quando é necessário realizar uma nova expansão do plano original. Isto é, em lugar de expandir um nó a partir da raiz, expandimos o sub-plano somente nas partes que sejam necessárias. Como resultado, pode ocorrer a propagação do efeito das mudanças em outras partes do plano.

3) *Replanejar. É* similar à re-expansão, ocorrendo porém em um nível mais elevado na árvore e provocando mudanças mais profundas e importantes, que também podem ser propagadas.

Em [Blum, 1995], os autores discutem uma abordagem condicional, orientada, onde consideram que, quando um plano falha, isto se deve à não ocorrência de determinada condição, que foi assumida como sendo verdadeira e que, portanto, deve ser alterada. Se todas as condições utilizadas pelo gerador de planos, da forma como foram usadas, são armazenadas, então a identificação e localização do erro consiste em localizar a violação detectada no processo. A informação armazenada irá dizer qual parte do plano é afetada e qual tipo de replanejamento será utilizado. Para a aplicação da estratégia de replanejamento apropriada, estas condições devem ser classificadas como:

- *Precondições.* São condições que devem ser verdadeiras antes da execução de uma ação.
- *Condições de expansão*. Influenciam a expansão do nó (ou submeta). As expansões podem ser *críticas*, quando causam sérias mudanças no plano, ou podem ser *não críticas*, quando afetam apenas uma pequena parte do plano.
- *Condições de decisão*. São condições que dependem do estado corrente e que são utilizadas para tomada de decisão quando existe mais de uma escolha. A escolha deve ser feita utilizando heurísticas que indiquem o caminho mais conveniente a ser escolhido, levando em consideração critérios de qualidade de serviços, disponibilidade e custo, que está intimamente ligado à quantidade de recursos que a aplicação desta ação agregou.

2.4 Planejamento e escalonamento

2.4.1 O problema de planejamento

Em [Blythe, J., Gil, Y., 2003], a modelagem de workflows como um problema de planejamento foi apresentada com o objetivo de automatizar ao

máximo a geração e execução de processos, principalmente os ligados a áreas onde há processamento distribuído e grande quantidades de recursos e dados.

Certas aplicações podem ser vistas como *workflows* complexos, onde devem ser realizadas várias transformações nos dados para a obtenção do produto final. As Figuras 5 e 6 exemplificam este processo, desde sua modelagem até a execução.

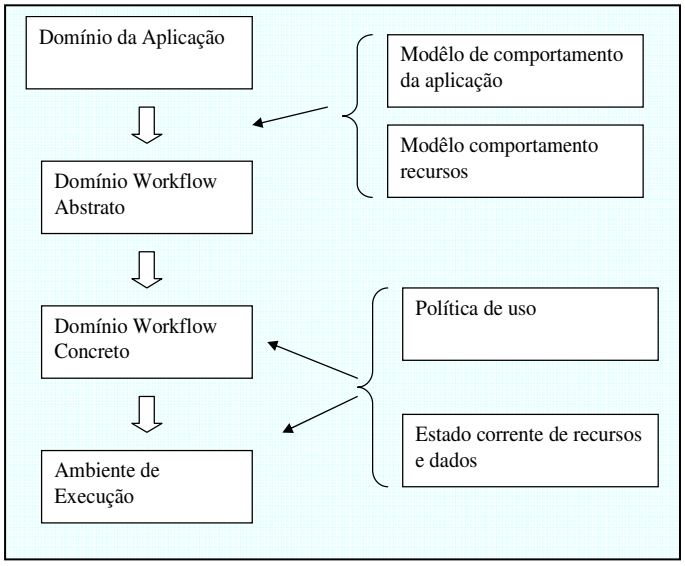

Figura 5 - Seqüência de geração de workflow

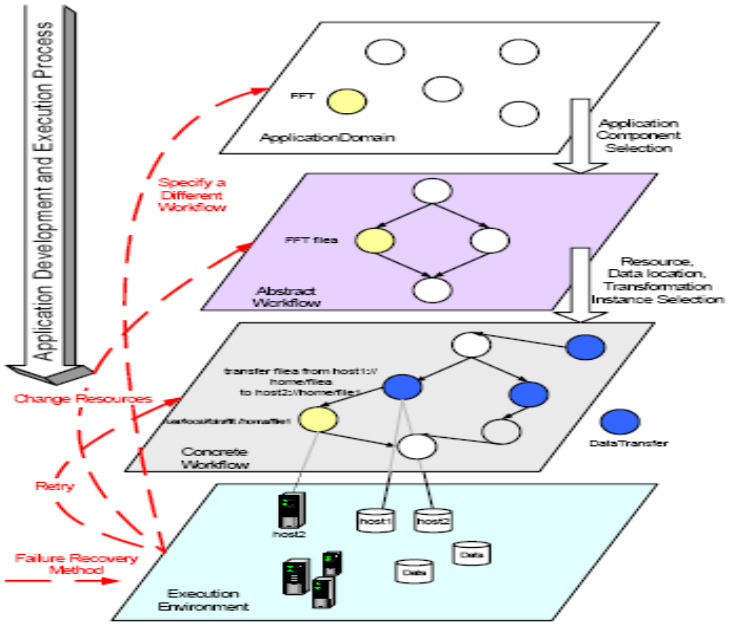

Figura 6 - Processo de desenvolvimento e execução [Blythe, J., Gil, Y., 2003]

Na geração do *workflow*, os componentes são modelados como transferências e os dados como operadores que representam a execução concreta de um componente em um local. As precondições representam ambos, as dependências entre os componentes, em função das entradas de dados e informações necessárias, além dos possíveis recursos necessários para a execução, incluindo o tipo de recurso.

Podemos dizer que um plano corresponde a um *workflow* concreto, . Alguns efeitos e precondições dos operadores capturam dados produzidos por componentes e as dependências entre estes dados. Como resultado deste processo, o planejador pode criar um *workflow* abstrato. O estado da informação utilizada pelo planejador inclui a descrição dos recursos disponíveis e dos arquivos relevantes que devem ser criados. As metas que fazem parte da entrada do workflow devem incluir:

- A descrição do metadado com a especificação da informação das requisições dos usuários e a localização desejada dos arquivos de saída;
- Componentes específicos para a execução das tarefas; e
- Dados intermediários produzidos.

A qualidade do plano deve ser considerada como a combinação da heurística utilizada, auxiliada pela escolha adequada dos componentes, e também pela busca exaustiva por soluções que minimizam o tempo total estimado de execução. Ambos os aspectos são considerados necessários, já que sem uma medida global de qualidade, várias escolhas locais "boas" podem ser combinadas num plano pobre, obtendo como resultado um plano inadequado.

2.4.2 O problema do escalonamento

Em [Fox, 1990], o problema de escalonamento é apresentado como "a capacidade de distribuir, durante um intervalo de tempo, as tarefas a serem executadas sobre recursos limitados, buscando otimizar tanto tempo quanto a utilização de recursos". Formalmente podemos definir um problema de escalonamento como um conjunto de tarefas $J=\{J_1, ..., J_n\}$ e um conjunto de recursos $R=\{R_1 ,..., R_n\}$, onde cada tarefa J_i consiste de um conjunto de operações $O^i=\{O^i_1, ...,O^i_n\}$ que devem ser realizadas entre o *tempo de prontificação*, Tp_i, e o *tempo de execução*, Te_i. Na execução de cada operação O^i_k é necessária a utilização de um conjunto de recursos $R^i_k \subseteq R$ durante determinado intervalo de tempo Tu_i. O *tempo de início* Ti^i_k da operação O^i_k indica quando a operação deve utilizar o recurso R^i_k. Este tipo de problema pode ser considerado como um *problema de satisfação de restrições* (PSR).

Os problemas de escalonamento podem ser caracterizados em função de parâmetros tais como:

- determinismo, frente a estocasticidade (duração das tarefas fixas);
- requisitos de tempo real (restrições de tempo frente a processos *off-line*);
- presença de restrições sobre os recursos;
- objetivos do escalonador (minimizar o espaço de solução, o custo, etc.);
- requisitos de otimização.

Independentemente dos parâmetros mencionados, existem duas grandes famílias de escalonamento, segundo os graus de liberdade em relação à demanda de recursos em um certo tempo:

- *Problemas de escalonamento puros*. A capacidade de cada recurso está definida sobre um certo número de intervalos temporais e o problema de escalonamento consiste em cobrir as demandas dos recursos e das atividades ao longo do tempo, sem exceder a capacidade disponível. Neste caso, conhecemos a priori que recurso vai ser utilizado por cada uma das atividades.
- *Problemas de distribuição de recursos*. Dispomos de um conjunto de operações e, para cada uma delas, de um conjunto de recursos idênticos (que realizam o mesmo tipo de operação), embora não equivalentes (podem requerer tempos distintos ou custos de processamento), que podem ser utilizados. Neste caso, não conhecemos a priori que recurso concreto vai ser utilizado em cada operação.

Dependendo do entorno, o tempo necessário para obter uma resposta de um problema de escalonamento varia, pois pode ser necessário modificar incrementalmente o escalonamento, ou seja, o escalonamento é *dinâmico*, principalmente devido a alterações no ambiente. Neste caso, existe a dificuldade adicional de que não se conhece a priori todas as restrições do problema, sendo necessário muitas vezes o re-escalonamento, quando estas alterações invalidam o resultado obtido.

Em problemas reais, dinâmicos e impredizíveis, a noção de escalonamento não está claramente diferenciada da noção de planejamento, o que implica em raciocinar sobre ações, estados, recursos e tempo de forma global. O objetivo do escalonamento é conduzir um sistema controlado a um estado determinado ou proporcionar um certo nível de serviço de forma permanente. Em qualquer caso, uma solução do problema se caracteriza por sua factibilidade e sua qualidade. Estas características permitem que as restrições possam ser classificadas como *factíveis* ou aquelas que devem necessariamente ser satisfeitas e as de *qualidade*, que podem ou não ser realizadas, permitindo ainda uma certa flexibilidade no seu cumprimento.

Outra característica de qualquer solução é a adequabilidade perante o contexto e a solução obtida. Quanto mais rapidamente se obtém uma solução, mais alta será a adequabilidade. Por outro lado, encontrar uma solução factível e com uma alta qualidade pode levar muito tempo. Portanto, podemos concluir que existe uma contradição entre estas duas características. Esta contradição é o principal problema do escalonamento dinâmico frente ao estático.

Uma vez fixado o horizonte temporal, devemos decidir quando deve ser realizado um novo processo de escalonamento. Esta escolha influência a adequabilidade da solução obtida. O escalonador deve recalcular novos valores quando novas condições invalidam as atuais, ou quando o horizonte temporal é alterado. Na maior parte do tempo, o escalonador deve resolver um problema dinâmico, no sentido que o novo problema é diferente do anterior.

2.4.3 Escalonamento e planejamento integrados

O problema da integração planejamento-escalonamento tem sido objeto de pesquisa recente [Gervasio, 1992], [Smith, 1993b], [Sadeh, 1996b], [Bartak, 1999], [Smith, 2000], [Bartak, 2000], [Garrido, 2000].

Os trabalhos de [Bertolotti, 1993], [Gervasio, 1992] apresentam e discutem problemas de planejamento e escalonamento, onde o foco principal passa a ser o de limitar a execução das ações no tempo, disponibilizando os recursos necessários, e considerando as restrições gerais do problema. De uma forma geral, o planejador considera as especificações das tarefas e recursos e gera um conjunto de operações que produzem o resultado desejado, considerando as restrições explicítas e os recursos necessários. Quando várias tarefas devem ser executadas conjuntamente, novas restrições devem ser adicionadas. Normalmente, durante o processo de planejamento não são considerados os aspectos de satisfação de restrições temporais, nem as necessidades de uso de recursos.

O escalonador, por outro lado, deve se preocupar em satisfazer tanto as restrições de ordenação explícitas, impostas pelos planos, quanto as implícitas, derivadas da disponibilidade dos recursos. O escalonador deve levar em

consideração outros tipos de restrições, tais como o tempo de duração das atividades, os intervalos de manutenção, os períodos de parada, etc... Normalmente, um sistema de escalonamento parte de um plano, que determina a seqüência de ações a serem realizadas e quando podem ser realizadas, e verifica a possibilidade da utilização compartilhada dos recursos, entre outras restrições, preocupando-se com critérios predeterminados de otimização. Alguns exemplos podem ser encontrados em [Aanen, 1988], [Khoshnevis, 1989], [Tonshoff, 1989], [Bossink, 1992], [Zhang, 1994], [Huang, 1995].

Na maior parte dos trabalhos, as atividades de planejamento e escalonamento se dão de forma independente e seqüencial, com pouca comunicação entre elas. Esta falta de coordenação conduz a planos com custos altos, a um tempo efetivamente alto para encerramento das tarefas, a atrasos devidos à indisponibilidades de recursos, e a restrições em relação à capacidade de coordenar de forma efetiva as operações locais e as realizadas externamente, quando necessário.

Em problemas reais, onde temos tanto restrições temporais quanto de uso de recursos, tradicionalmente numa primeira fase geramos o plano e posteriormente aplicamos o escalonador para garantir a realização do plano e a satisfação das restrições temporais. Na arquitetura apresentada a seguir, o processo se divide em etapas sucessivas [Srivastava, 1999]. Na etapa de planejamento, obtém-se um plano, composto de uma seqüência ordenada de ações. Na segunda etapa, executa-se o escalonamento dos recursos, procurando otimizar a distribuição, e garantindo o cumprimento das restrições existentes.

Assim, teoricamente, temos um plano que resolve o problema, embora os processos sejam independentes e sem nenhum tipo de relação entre eles durante sua execução. O inconveniente apresentado por esta abordagem está no fato das decisões serem tomadas localmente, podendo gerar problemas ou inconvenientes. Estas soluções foram apresentadas em [Drummond, 1990], [Mitchell, 1990]. Em [Martin, 1990], [Olawsky, 1990] as soluções, embora reativas, são atrasadas até que determinadas condições sejam cumpridas. Abordagens reativas também têm sido empregadas para incorporar informações sobre as decisões do escalonador

em tempo de execução, procurando desta forma otimizar e trabalhar em domínios dinâmicos [Ow, 1988], [Prosser, 1989], [Zweben, 1993], [Zilberstein, 2000].

Um enfoque oposto ao apresentado anteriormente consiste em integrar os processos de planejamento e escalonamento em uma única arquitetura, de forma que ambos trabalhem em conjunto: o planejador gera planos parciais, que o escalonador valida ou descarta. Assim, em caso de necessidade, o plano parcial pode ser modificado sem a necessidade de gerar um novo plano.

Em [Sadeh, 1996] e [Srivastava, 1999] esta integração é realizada de três formas distintas:

1) Desenvolvendo um processo de planejamento que gere todos os planos possíveis que são solução do problema e um sistema de escalonamento que selecione dinamicamente, entre estas alternativas, a mais adequada. Esta abordagem sobrecarrega tanto o planejador, que deve definir todos os planos possíveis criando uma espécie de biblioteca, quanto o escalonador, que deve tratar todos os planos possíveis que são obtidos.
2) Desenvolvendo um processo de planejamento reativo em tempo real, onde são levadas em conta as informações dinâmicas sobre a disponibilidade de recursos.
3) Desenvolvendo um processo de planejamento distribuído e hierarquizado com objetivo de reduzir a complexidade do processo, mediante uma subdivisão das decisões integrando com o sistema de planejamento/escalonamento em fases e com decisões localizadas.

Uma integração eficiente de ambos os sistemas requer, num primeiro nível, que o conjunto de ações seja determinado e, num segundo nível, que as ações sejam instanciadas por um planejador/escalonador. Os processos de planejamento e de escalonamento passam desta forma a serem simultâneos.

Existem diversos precedentes de aproximação entre planificação e escalonamento ([Smith, 1996], [Laliberty, 1996], [Smith, 1993], [Tsamardinos, 1988], etc.). Grande parte das aproximações estão baseadas em um tipo de planificação temporal, que vem a ser a satisfação de restrições temporais por parte das ações do plano. Este procedimento dá lugar a uma seqüenciação mais restrita

das ações do plano. Em algums sistemas de planejamento, como O-Plan [Currie, 1991], SIPE [Wilkins, 1995] e ParcPlan [Kholy, 1996]), observamos uma certa integração entre o planejamento e o escalonamento. Da mesma forma, os planejadores temporais, como DEVISER [Vere, 1983] e IxTet [Gallab, 1994]), são capazes de realizar inferências envolvendo restrições temporais.

As tecnologias existentes atualmente tanto para planejamento quanto para escalonamento não permitem uma integração efetiva de ambos os processos. Existem diferenças na estrutura do problema e nos domínios associados. Em [Smith, 2000] é feito um estudo sobre as diferentes técnicas de planejamento e escalonamento existentes, com o foco nas similaridades, diferenças e limitações.