Escola de Computação'2000 - IME-USP

Sistemas de Tempo Real: Sistemas Operacionais

Jean-Marie Farines
Joni da Silva Fraga
Rômulo Silva de Oliveira

LCMI - Laboratório de Controle e Microinformática
DAS - Departamento de Automação e Sistemas
UFSC - Universidade Federal de Santa Catarina

Suportes para Aplicações de Tempo Real

- Aplicações são construídas a partir
  dos serviços oferecidos por um sistema operacional
- O atendimento dos requisitos temporais depende
  não somente do código da aplicação,
  mas também da colaboração do sistema operacional
- No sentido de permitir **previsibilidade**
  ou pelo menos um **desempenho satisfatório**

Suportes para Aplicações de Tempo Real

- Muitas vezes os requisitos temporais da aplicação
  são tão rigorosos que
  o sistema operacional é substituído
  por um **simples núcleo de tempo real**

- Não inclui serviços como
  - sistema de arquivos ou
  - gerência sofisticada de memória
- Núcleos de tempo real oferecem uma funcionalidade mínima,
- Mas são capazes de apresentar
  excelente comportamento temporal

Suportes para Aplicações de Tempo Real

- Sistemas operacionais convencionais são construídos com o objetivo
  de apresentar um bom comportamento médio
- Distribuem os recursos do sistema de forma justa
  entre as tarefas e os usuários
- Não existe uma preocupação com previsibilidade temporal
- Mecanismos como
  - Caches de disco
  - Memória virtual
  - Fatias de tempo do processador
- melhoram o desempenho médio do sistema
- mas tornam mais difícil fazer afirmações sobre
  os tempos de uma tarefa em particular

Sistema Operacional de Tempo Real

- Aplicações com restrições de tempo real
  - menos interessadas em uma distribuição uniforme dos recursos
  - mais interessadas em atender requisitos
    tais como períodos de ativação e deadlines
- Sistemas operacionais de tempo real
  - atenção é dedicada ao comportamento temporal
  - serviços são definidos não somente em termos funcionais
    mas também em termos temporais

- **Plataforma alvo** (*target system*)
  - Hardware e o SOTR onde a aplicação vai executar quando pronta
- **Plataforma de desenvolvimento** (*host system*)
  - Hardware e o SO onde o sistema é desenvolvido

Aspectos Funcionais de um SOTR

- Como qualquer sistema operacional
  SOTR procura tornar a utilização do computador
  - mais eficiente
  - mais conveniente
- Facilidades providas por um sistema operacional de propósito geral
  são bem vindas em um SOTR

- Aplicações de tempo real são usualmente organizadas na forma de
  várias *threads* ou tarefas concorrentes
- Logo, um requisito básico para os sistemas operacionais de tempo real
  é oferecer suporte para tarefas e *threads*

Tarefas e Threads

- Tarefas ou processos são abstrações que incluem
  - um espaço de endereçamento próprio (possivelmente compartilhado)
  - um conjunto de arquivos abertos
  - um conjunto de direitos de acesso
  - um contexto de execução formado pelos registradores do processador
  - vários outros atributos
- *Threads* são tarefas leves
  - únicos atributos são aqueles associados com o contexto de execução
- Chaveamento entre duas *threads* de uma mesma tarefa
  é muito mais rápido
  que o chaveamento entre duas tarefas
- Qualquer SO provê tarefas
  é bom ter threads também

Comunicação entre Tarefas e entre Threads

- Uma aplicação tempo real é tipicamente um programa concorrente
  - Formado por tarefas e/ou *threads*
  - que se comunicam e se sincronizam
- A literatura sobre sistemas operacionais trata este assunto
- Existem duas grandes classes de soluções p/ programação concorrente
  - Troca de mensagens
  - Variáveis compartilhadas
- A correta programação da comunicação e sincronização das tarefas
  - Garante o seu comportamento funcional
  - Mas não o seu comportamento temporal
- Algumas soluções de escalonamento tempo real resolvem os dois
  - Escalonamento baseado em executivo cíclico
  - Não existe a necessidade de mecanismos explícitos de sincronização

Instalação de Tratadores de Dispositivos

- Sistemas de tempo real lidam com periféricos especiais, diferentes tipos de sensores e atuadores
  - Automação industrial
  - Controle de equipamentos em laboratório
- Projetista da aplicação deve ser capaz de
  - desenvolver os seus próprios tratadores de dispositivos (*device drivers*)
  - incorpora-los ao sistema operacional
- Muitas vezes a aplicação e o periférico estão fortemente integrados
  - código da aplicação confunde-se com o código do tratador do dispositivo
  - acontece no contexto dos sistemas embutidos (*embedded systems*)
- SOTR deve permitir a aplicação instalar os seus próprios tratadores de interrupções

Temporizadores

- Aplicações precisam realizar operações que manipulam tempo
  - Ler a hora com o propósito de atualizar um histórico
  - Realizar determinada ação a cada X unidades de tempo
  - Realizar uma ação depois de Y unidades de tempo a partir de agora
  - Realizar determinada ação a partir do instante absoluto de tempo Z
- SOTR deve ofereçer um conjunto de serviços que atenda estas necessidades
- Tipicamente o sistema possui pelo menos um temporizador (*timer*) implementado em hardware
  - Gera interrupções com uma dada frequência
  - SOTR utiliza este temporizador para criar temporizadores lógicos
- Resolução ou granularidade
  - Dada pela frequência do temporizador do hardware

Aspectos Temporais

- Aplicação e SOTR compartilham os mesmos recursos do hardware
- Comportamento temporal do SOTR
  afeta o comportamento temporal da aplicação
- Por exemplo,
  rotina do sistema operacional que trata as interrupções do timer
- O projetista da aplicação pode ignorar completamente
  a função desta rotina
- Mas não pode ignorar o seu efeito temporal,
  a interferência que ela causa na execução da aplicação
- Solicitar um serviço ao SO através de chamada de sistema significa:
  - processador **será ocupado** pelo código do sistema operacional
  - capacidade da aplicação atender seus deadlines **passa a depender**
    da capacidade do SO em fornecer o serviço solicitado
    em um tempo que não inviabilize aqueles deadlines

Aspectos Temporais

- Teoria de tempo real, descrita no capítulo anterior, é recente
- Referência mais antiga seja sempre [LiL73]
- Somente na década de 90 os modelos de tarefas suportados
  foram estendidos a ponto de tornarem-se verdadeiramente úteis
- Estes avanços da teoria estão sendo
  gradativamente absorvidos pelos desenvolvedores de SOTR
- Ainda existe uma distância
  entre a teoria de escalonamento
  e a prática no desenvolvimento
  de sistemas de tempo real

- De um lado a teoria buscando a previsibilidade
- De outro a prática fazendo "o que é possível"
  nos ambientes computacionais existentes

Limitações dos SO de Propósito Geral

- Diversas técnicas populares em SOPG
  são especialmente problemáticas
  quando as aplicações possuem requisitos temporais
- Mecanismo de memória virtual é capaz de gerar grandes atrasos
- Mecanismos tradicionais usados em sistemas de arquivos,
  fazem o tempo para acessar um arquivo variar muito
  - ordenar a fila do disco para diminuir o tempo médio de acesso
- Aplicações de tempo real procuram minimizar o efeito negativo
- Desativa o mecanismo sempre que possível
- Usa o mecanismo apenas em tarefas sem requisitos temporais rigorosos
  - acesso a disco feito por tarefas sem requisitos temporais

Limitações dos SO de Propósito Geral

- Todos os sistemas operacionais
  desenvolvidos ou adaptados para tempo real
  mostram grande preocupação
  com a divisão do tempo do processador entre as tarefas
- Entretanto, o processador é apenas um recurso do sistema
- Memória, periféricos, controladores
  também deveriam ser escalonados
  visando atender os requisitos temporais da aplicação
- Muitos sistemas ignoram isto
  - Tratam os demais recursos da mesma maneira empregada por um sistema operacional de propósito geral

Limitações dos SO de Propósito Geral

- Tipicamente qualquer sistema operacional dispõe de escalonamento baseado em prioridades
- Entretanto, a maioria dos sistemas operacionais de propósito geral inclui mecanismos que reduzem automaticamente a prioridade na medida que a tarefa consome tempo de processador
  - Envelhecimento ou "aging"
- Mecanismo utilizado para
  - favorecer as tarefas com ciclos de execução menor
  - diminuir o tempo médio de resposta no sistema
- Em sistemas de tempo real
  justa distribuição de recursos entre as tarefas é menos importante do que o atendimento dos requisitos temporais

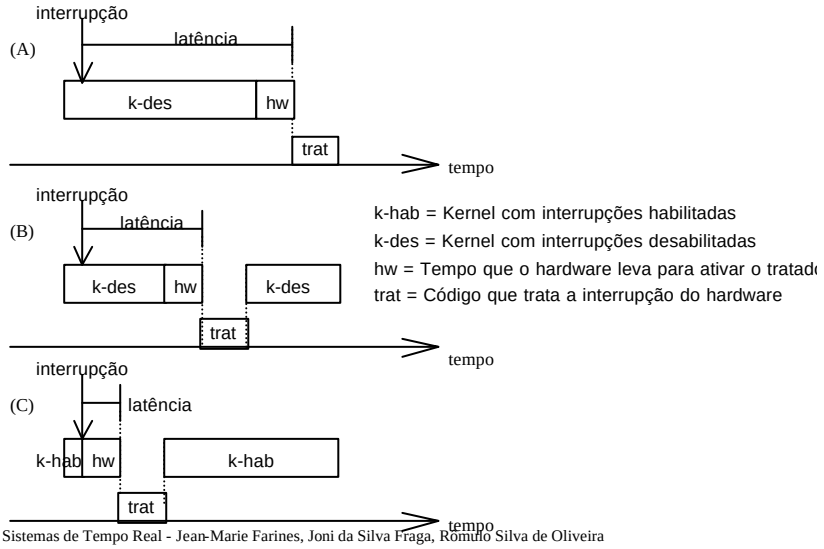
Limitações dos SO de Propósito Geral - Métricas

- Fornecedores de SOTR costumam divulgar métricas
- Para mostrar como o sistema suporta aplicações de tempo real
- Estas métricas refletem a prática da construção de aplicações TR
  - Ligadas à desempenho
- Uma métrica muito utilizada é
  o tempo para chaveamento entre duas tarefas
- Este tempo inclui
  - salvar os registradores da tarefa que está executando
  - carregar os registradores com os valores da nova tarefa
- Não inclui o tempo necessário para decidir qual tarefa vai executar
  - depende do algoritmo de escalonamento

Limitações dos SO de Propósito Geral - Métricas

- Outra métrica é a latência
  até o inicio do tratador de uma interrupção do hardware
- Eventos importantes e urgentes no sistema
  serão sinalizados por interrupções
- Importante iniciar rapidamente o tratamento destas interrupções
- Na análise de escalonabilidade
  - Tratador de interrupções corresponde a tarefa com a prioridade mais alta
  - Gera interferência sobre as demais tarefas
  - Latência de interrupção equivale ao *release jitter* desta tarefa especial
- Tempo definido pela forma como o kernel foi programado

Limitações dos SO de Propósito Geral - Métricas



Limitações dos SO de Propósito Geral - Métricas

- Métricas como o tempo de chaveamento e o de latência são úteis
  - Quanto menor elas forem em um dado SOTR, tanto melhor
  - São relativamente fáceis de medir
  - Valores necessários no momento de aplicar os testes de escalonabilidade
- Não são as únicas responsáveis pelos tempos de resposta
  - Diversas tarefas e interrupções do sistema causam interferências
  - Conflitos decorrentes de recursos compartilhados
    tanto a nível de aplicação como a nível de kernel
    causam situações de bloqueio e de inversão de prioridades
  - Todos estes fatores devem ser computados
- Um dos maiores problemas atualmente
  para implementar os algoritmos descritos no capítulo 2
  é o desconhecimento dos conflitos decorrentes
  de recursos compartilhados a nível de kernel

Teoria de Escalonamento e Sistema Operacional

- Abordagem de escalonamento determinada pela natureza da aplicação
  - Crítica ou não, carga estática ou não, etc
- Questão fundamental para quem vai usar um SOTR é
  - determinar sua capacidade de suportar a abordagem de escalonamento
- Escalonamento baseado em prioridades preemptivas é suficiente
  - desde que os atrasos e bloqueios do SOTR sejam conhecidos
- Maior obstáculo à aplicação da teoria de escalonamento é
  dificuldade em determinar os tempos máximos de execução
- Dependem de vários fatores como
  - fluxo de controle
  - arquitetura do computador (*cache*, *pipeline*, etc)
  - velocidade de barramento e do processador, etc
- Existam ferramentas experimentais nesta área,
  ainda não existem ferramentas com qualidade suficiente

Teoria de Escalonamento e Sistema Operacional

- Existem alguns caminhos para contornar este problema
- Em tempo de projeto, quando o código ainda não existe,
  é possível estimar os tempos de execução das rotinas
- Usar estas estimativas como dados de entrada
  nas equações do capítulo 2
  - Os resultados são estimativas
- Permite detectar durante o projeto problemas futuros
  com respeito aos tempos de resposta das tarefas
- Detectar a necessidade de alterações antes de iniciar a programação
  muito melhor do que
  fazer alterações depois que tudo já estiver programado

Teoria de Escalonamento e Sistema Operacional

- Uma vez que a aplicação esteja programada possível analisar se
  as estimativas usadas em tempo de projeto foram adequadas
- Embora existam ferramentas que fazem isto automaticamente
  são ainda projetos acadêmicos,
  sem a qualidade necessária para utilização em projetos

- Uma alternativa é medir os tempos de execução
- Não existe a garantia de que o pior caso apareça nas medições
- Elas fornecem uma boa idéia para o tempo de execução da tarefa
- Uma margem de segurança pode ser associada

Teoria de Escalonamento e Sistema Operacional

- Grande obstáculo à aplicação da teoria de escalonamento é obter os atrasos e bloqueios associados com o SOTR
  - em função de chamadas de sistema
  - interrupções de hardware
  - acesso a periféricos, etc
- Análise de escalonabilidade requer detalhes do SOTR
  - Maioria das vezes não são disponibilizados pelo fornecedor
  - Este quadro deverá mudar lentamente
- Se aplicação é do tipo tempo real brando (*soft real-time*)
  - Projetista escolhe um sistema operacional com boas propriedades
  - Escalonamento baseado em prioridades preemptivas
  - Kernel que executa com interrupções habilitadas
  - Chaveamento de contexto rápido
  - Baixa latência de interrupção

Tipos de Suportes para Tempo Real

- A diversidade de aplicações gera uma diversidade de necessidades
- Resulta em um leque de soluções com respeito aos suportes
- Com diferentes tamanhos e funcionalidades

- Podemos classificar os suportes de tempo real em dois tipos:
  - Núcleos de tempo real (NTR)
  - Sistemas operacionais de tempo real (SOTR)

- NTR consiste de um pequeno kernel
  - com funcionalidade mínima
  - mas excelente comportamento temporal
  - indicada para, por exemplo, o controlador de uma máquina industrial

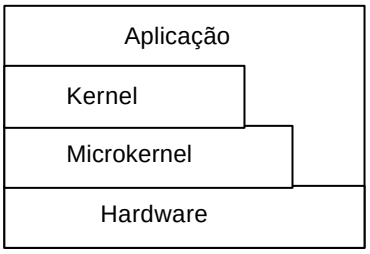
Tipos de Suportes para Tempo Real

- SOTR é um sistema operacional completo
  - funcionalidade típica de propósito geral
  - mas cujo kernel foi adaptado para melhorar o comportamento temporal
  - qualidade temporal do kernel adaptado varia de sistema para sistema
    - alguns são completamente reescritos para tempo real
    - outros recebem apenas algumas poucas otimizações

	Funcionalidade mínima	Funcionalidade completa
Previsibilidade maior	Núcleo de Tempo Real	Futuro...
Previsibilidade menor	Qualquer Núcleo Simples	Sistema Operacional Adaptado

Microkernel

- Sistemas podem ser organizados em camadas
- Subindo na estrutura de camadas
  - os serviços tornam-se mais sofisticados
  - o comportamento temporal menos previsível
- Aplicação tem a sua disposição uma gama completa de serviços
- Quando os requisitos temporais da aplicação aumentam
  - pode acessar diretamente o microkernel
  - e até mesmo o hardware
- Apropriado para sistemas onde apenas uma aplicação é executada



Escolha de um Suporte de Tempo Real

- Difícil comparar diferentes SOTR
  - Diferentes abordagens de escalonamento
  - Desenvolvedores de SOTR publicam métricas diferentes
  - Desenvolvedores de SOTR não publicam todas as métricas
  - Detalhes internos sobre o kernel não estão normalmente disponíveis
  - Métricas fornecidas foram obtidas em plataformas diferentes
  - Conjunto de ferramentas para desenvolvimento que é suportado varia
  - Ferramentas para monitoração e depuração das aplicações variam
  - Linguagens de programação suportadas em cada SOTR são diferentes
  - Conjunto de periféricos suportados por cada SOTR varia
  - Conjunto de plataformas de hardware suportados varia
  - Cada SOTR possui um esquema para a incorporação de *device-drivers*
  - Possuem diferentes níveis de conformidade com os padrões
  - Política de licenciamento e o custo associado variam

POSIX para Tempo Real

- Posix é um padrão para sistemas operacionais
  - baseado no Unix
  - criado pela IEEE (Institute of Electrical and Eletronic Engineers)
- Posix define as interfaces do sistema operacional
  mas não sua implementação
  - Posix API (*Application Programming Interface*)
- Sistemas operacionais de tempo real possuem uma API proprietária
  - Aplicação fica amarrada aos conceitos e às primitivas
    do sistema em questão
- Usando um SOTR que é compatível com Posix,
  - Aplicação fica amarrada aos conceitos e às primitivas do Posix
- Muitos SOTR atualmente já suportam a API do Posix

Linux para Tempo Real

- Linux é um sistema operacional com fonte aberto, estilo Unix,
  - Inclui multiprogramação, memória virtual, bibliotecas compartilhadas, protocolos de rede TCP/IP, etc
- Uma descrição completa do Linux não cabe neste livro
- Linux convencional segue o estilo de um kernel Unix tradicional
  - kernel monolítico, não é baseado em microkernel
  - não é apropriado para a maioria das aplicações de tempo real
- Kernel do Linux possui um recurso que facilita sua adaptação
  - Aceita "módulos carregáveis em tempo de execução"
  - Podem ser incluídos e excluídos do kernel sob demanda
- Soluções de escalonamento tempo real podem ser incluidas no kernel
- http://www.linux.org/projects/software.html
  - 3 projetos envolvendo adaptações do Linux para tempo real
- LINUX-SMART, desenvolvido no Brasil (IME-USP)

Conclusões

- Área de sistemas operacionais de tempo real é muito dinâmica
- Novos sistemas ou novas versões dos sistemas existentes
  são apresentadas a todo momento

- Mensagem central:
  - O comportamento temporal da aplicação tempo real depende
    tanto da aplicação em si quanto do sistema operacional
  - Desta forma, a seleção do SOTR a ser usado depende fundamentalmente
    dos requisitos temporais da aplicação em questão
- Não existe um SOTR melhor ou pior para todas as aplicações
- A diversidade de aplicações tempo real existente gera
  uma equivalente diversidade
  de sistemas operacionais de tempo real