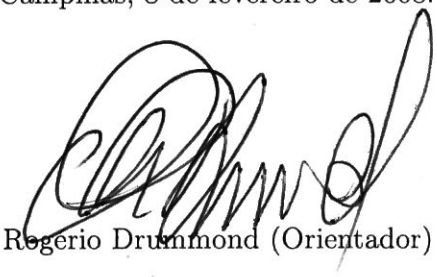
Uma abordagem interdisciplinar para agendamento em grupo

Este exemplar corresponde à redação final da Dissertação devidamente corrigida e defendida por Fábio Henrique da Silva e aprovada pela Banca Examinadora.

Campinas, 8 de fevereiro de 2008.



Rogério Drummond (Orientador)

Dissertação apresentada ao Instituto de Computação, UNICAMP, como requisito parcial para a obtenção do título de Mestre em Ciência da Computação.

FICHA CATALOGRÁFICA ELABORADA PELA BIBLIOTECA DO IMECC DA UNICAMP
Bibliotecária: Maria Júlia Milani Rodrigues – CRB8a / 2116

Silva, Fábio Henrique da
Si38a Uma abordagem interdisciplinar para agendamento em grupo / Fábio Henrique da Silva -- Campinas, [S.P. :s.n.], 2008.

Orientador : Rogério Drummond
Dissertação (mestrado) - Universidade Estadual de Campinas, Instituto de Computação.

1. Protocolos. 2. Otimização. 3. Inteligência artificial. 4. Interfaces. I. Drummond, Rogério. II. Universidade Estadual de Campinas. Instituto de Computação. III. Título.

Título em inglês: An interdisciplinary approach for group scheduling.

Palavras-chave em inglês (Keywords): 1. Protocols. 2. Optimization. 3. Artificial intelligence. 4. Interfaces.

Área de concentração: Sistemas de Informação

Titulação: Mestre em Ciência da Computação

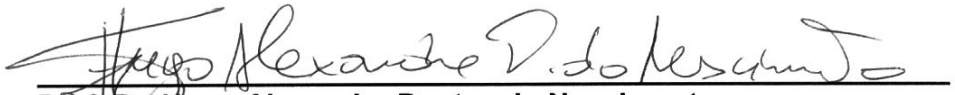
Banca examinadora: Prof. Dr. Hugo Alexandre Dantas do Nascimento (INF-UFG)
Profa. Dra. Maria Cecília Calani Baranauskas (IC-UNICAMP)
Prof. Dr. Cid Carvalho de Souza (IC-UNICAMP)

Data da defesa: 08/02/2008

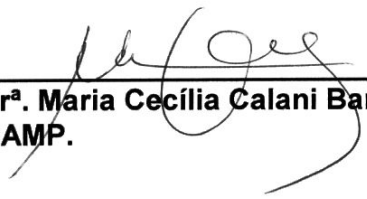
Programa de Pós-Graduação: Mestrado em Ciência da Computação

TERMO DE APROVAÇÃO

Dissertação Defendida e Aprovada em 08 de fevereiro de 2008, pela Banca examinadora composta pelos Professores Doutores:



Prof. Dr. Hugo Alexandre Dantas do Nascimento
Instituto de Informática/Universidade Federal de Goiás.



Profª. Drª. Maria Cecília Calani Baranauskas
IC/UNICAMP.

Prof. Dr. Rogério Drummond Burnier Pessoa de Mello Filho
IC/UNICAMP.

Instituto de Computação
Universidade Estadual de Campinas

Uma abordagem interdisciplinar para agendamento em grupo

Fábio Henrique da Silva

Janeiro de 2008

Banca Examinadora:

- Rogério Drummond (Orientador)
- Hugo A. D. do Nascimento
  Instituto de Informática – UFG
- Maria Cecília Calani Baranauskas
  Instituto de Computação – Unicamp
- Cid Carvalho de Souza (Suplente)
  Instituto de Computação – Unicamp

Resumo

Agendamento em grupo é uma atividade essencial e contínua. As complexidades na conciliação de agendas de diversas pessoas fazem da administração do tempo um desafio. Ele envolve sofisticados mecanismos de comunicação para suportar compartilhamento de calendários e negociações de agendamento entre usuários de diversos sistemas, compreendendo também a necessidade de reagendamento de compromissos já marcados em favor de outros. Entretanto, faltam soluções que facilitem a coordenação de situações complexas como estas. Com base em uma revisão das abordagens passadas para este problema e seus sucessos e falhas, uma nova abordagem, interdisciplinar, é apresentada para agendamento em grupo. O desenvolvimento dos protocolos de compartilhamento de calendários e agendamento é analisado, bem como a aplicação de promissoras técnicas nos campos de otimização, inteligência artificial e interfaces. Em direção a uma nova geração de ferramentas de agendamento em grupo mais inteligentes e poderosas que trarão ganhos de produtividade minimizando esforços de coordenação, uma aplicação multidisciplinar de conhecimentos destas áreas é proposto. Extensões para os atuais protocolos de negociação de agendamento são descritas, e as iniciativas de desenvolvimento do iScheduler, uma solução que aplica as melhores práticas identificadas neste estudo, são relatadas.

Abstract

Group scheduling is an essential and continuous activity. Complexities on conciliating schedules of several people make time management a challenge. It involves sophisticated communication mechanisms to support calendar sharing and scheduling negotiation among users of diverse systems, also comprising the need of rescheduling meetings in favor of others. However, there is a lack of solutions to ease the coordination of more complex situations like these. Based on a review of past approaches for this problem and their successes and failures, a new, interdisciplinary approach is presented for group scheduling. The development of calendaring and scheduling protocols is reviewed, as well as the application of promising techniques on the fields of optimization, artificial intelligence and interfaces. Towards a new generation of more intelligent and powerful group scheduling tools that will increase productivity by minimizing coordination efforts, a multidisciplinary application of knowledge from these areas is proposed. Extensions for present scheduling negotiation protocols are described, and the initiatives on the development of iScheduler, a solution that applies the best practices identified on this study, are reported.

Agradecimentos

A Deus, por tudo.

Aos meus pais, que me ensinaram que o conhecimento é o investimento mais seguro, pois não pode ser perdido. Pelo incentivo, e por quem sou.

A Jhoyce, pelo carinho e apoio incondicionais, e pelas horas de convívio suprimidas.

Ao professor Rogério Drummond, pela amizade, confiança, paciência para discutir novas idéias, e liberdade que me concedeu para trilhar caminhos neste trabalho de pesquisa. Aos demais companheiros de trabalho do Laboratório A-HAND – Elton, Pedro, Sônia e Wilson – pelos agradáveis momentos de trabalho conjunto.

Aos professores Cid Carvalho de Souza e Hans Liesenberg, por terem aceitado o convite para participar da banca de qualificação, e pelos relevantes comentários com respeito à proposta de trabalho então apresentada.

A todos os pesquisadores e desenvolvedores que contribuíram para o desenvolvimento deste projeto: Dave Thewlis (CalConnect Consortium), pela disponibilidade em circular minhas questões entre os membros do consórcio; Mikael Grev (MiG InfoCom), pela licença acadêmica do componente MiG Calendar; Gunnar Klau (Free University of Berlin) e Joe Marks (Mitsubishi Electric Research Laboratories), pela concessão e apoio no uso do framework HuGS; Bernard Desruisseaux (Oracle) e demais participantes das listas de discussão de protocolos, pelos comentários a respeito de nossas propostas; e Ben Fortuna (iCal4j), pelas interações relativas à utilização de sua biblioteca.

Aos amigos da UniSoma; em especial ao professor Miguel Taube Netto e a Eduardo Milanez, sem cujos apoios o desenvolvimento deste trabalho não seria possível.

Enfim, a todos os meus familiares e amigos, pelo apoio e expectativa com relação à conclusão desta etapa.

Conteúdo

Resumo **ix**

Abstract **xi**

Agradecimentos **xiii**

1 Introdução **1**
1.1 Motivação 1
1.2 Objetivos 3
1.3 Organização 4

2 Conceitos **5**
2.1 Otimização combinatória e Inteligência artificial 6
2.2 Redes e protocolos 9
2.2.1 O desenvolvimento de padrões e a IETF 10
2.3 Sistemas distribuídos 11
2.4 Agendamento em grupo 12
2.4.1 Formalizando agendamento distribuído 13
2.4.2 Reagendamento 14
2.5 Interfaces 15

3 Revisão das abordagens para agendamento em grupo **17**
3.1 A primeira geração de sistemas de agendamento 18
3.2 O estabelecimento de protocolos de comunicação 19
3.2.1 iCalendar 20
3.2.2 iTIP 24
3.2.3 iMIP 32
3.3 Os sistemas de segunda geração e a evolução dos protocolos 33
3.3.1 Um protocolo de transporte em tempo real para iTIP 35
3.3.2 Evolução dos protocolos já existentes 37
3.4 Limitações dos protocolos 38

3.5 Agendamento em grupo como um problema de otimização 40
3.5.1 Timetable Rearrangement 42
3.6 Agendamento em grupo como um problema de inteligência artificial 50
3.6.1 Um modelo simples de preferências dos usuários 51
3.6.2 Iniciativas contemporâneas 54
3.7 Agendamento em grupo como um problema de interfaces 54
3.8 Abordagens inovadoras 55
3.8.1 LookOut 57
3.8.2 HuGS 58

4 Uma nova abordagem **61**
4.1 Agendamento em grupo, um desafio multidisciplinar 61
4.2 Introduzindo novos conceitos 62
4.2.1 Tarefas líquidas 62
4.2.2 Eventos imprecisos e eventos alternativos 64
4.3 Especificação de extensões de protocolos 65
4.3.1 Eventos imprecisos e alternativos para iCalendar e iTIP 65
4.3.2 Draft de especificação das extensões 71
4.4 Esboçando os sistemas de terceira geração 72

5 iScheduler **73**
5.1 Arquitetura 73
5.2 Implementações 74
5.2.1 Biblioteca para extensões do padrão iCalendar 74
5.2.2 Algoritmo interativo para Timetable Rearrangement 75
5.3 Resultados 80

6 Considerações finais **83**
6.1 Contribuições 85
6.2 Trabalhos futuros 85

A Especificação das extensões para iCalendar e iTIP **87**

B Detalhamento técnico das implementações **141**
B.1 Extensões do padrão iCalendar 141
B.2 Algoritmo interativo para Timetable Rearrangement 143

Bibliografia **147**

Lista de Tabelas

3.1 Métodos iTIP 25
3.2 Componentes para os quais os métodos iTIP são válidos 26

Lista de Figuras

1.1 Esboço de uma ferramenta de agendamento mais inteligente 2

2.1 Topologia de espaço de estados 8
2.2 Modelos de referência OSI e TCP/IP 10
2.3 Alguns dos protocolos mais comuns da Internet 11
2.4 Temporização de eventos com relógios lógicos 12

3.1 Compartilhamento de calendários e agendamento em grupo via soluções proprietárias 18
3.2 A integração de sistemas heterogêneos prevista pelo grupo CalSch 19
3.3 Exemplo de objeto iCalendar 21
3.4 Exemplo de evento recorrente 22
3.5 Exemplo de *to-do* 23
3.6 Exemplo de *free-busy time* 23
3.7 Exemplo de *journal entry* 24
3.8 Arquitetura iTIP 25
3.9 Exemplo de método PUBLISH para um evento 26
3.10 Exemplo de método PUBLISH para diversos componentes 27
3.11 Exemplo de método REQUEST 28
3.12 Exemplo de método REPLY 29
3.13 Exemplo de método CANCEL 30
3.14 Exemplo de métodos COUNTER e DECLINECOUNTER 31
3.15 Exemplo de método REFRESH 32
3.16 Exemplo de método REQUEST em resposta a REFRESH 33
3.17 Apple iCal 34
3.18 Integração entre dispositivos e aplicações distintas com CalDAV 37
3.19 A flexível estrutura hierárquico-recursiva da ontologia CoolAgent 39
3.20 Exemplo de papéis dos participantes de um evento iCalendar 39
3.21 O uso de gráficos de Gantt na interface do Microsoft Project 40
3.22 Representação gráfica do agendamento $T(5)$ 43
3.23 Representação gráfica do agendamento $T(6)$ 45

3.24 Uma árvore de busca para RTR(x,X) 46
3.25 Algoritmo com heurística para TR 47
3.26 Operações utilizadas na busca A* 48
3.27 Árvore de busca A* 50
3.28 A base de dados de senso comum do SensiCal em ação 51
3.29 Modelo de preferências de agendamento 52
3.30 Agendamento no sistema LookOut 57
3.31 Aplicação interativa HuGS para jobshop 59

4.1 Intersecção de áreas de conhecimento nas soluções para agendamento . . . 61
4.2 Exemplo de evento impreciso simples 66
4.3 Exemplo de evento impreciso recorrente com vários períodos de tempo . . 67
4.4 Exemplo de evento impreciso representando tarefa líquida 68
4.5 Exemplo de eventos alternativos 69

5.1 Arquitetura do iScheduler 73
5.2 Execução do algoritmo com instância $T(5)$ da figura 3.22 77
5.3 Solução ótima $T(6)$ da figura 3.23 77
5.4 Mensagens de aviso para movimentos shift e change inválidos 78
5.5 Solução ótima para uma instância da versão generalizada de TR 79
5.6 Agendamento privilegiando horários mais cedo 81
5.7 Agendamento privilegiando horários mais tarde 81

6.1 Os “dois elefantes” e o desenvolvimento de padrões 84

B.1 Hierarquia de classes das implementações realizadas 146

Capítulo 1

Introdução

Marcar compromissos, fazer reservas de salas e equipamentos e designar tarefas a pessoas são algumas das atividades de agendamento em grupo desempenhadas nas mais diversas situações cotidianas.

Como uma atividade essencial e contínua, a conciliação das agendas de diversas pessoas é um grande desafio para a administração do tempo, cada vez mais escasso. A complexidade de administrá-lo tem crescido de forma inversamente proporcional à sua disponibilidade, e faltam ferramentas adequadas para lidar com a infinidade de limitações, preferências, demandas e prazos, entre outros aspectos, não raro conflitantes entre si.

Seria possível marcar uma consulta médica sem perder a reunião? Como atribuir a pesada carga didática do departamento entre os poucos professores sem desagradá-los? Quais compromissos devem ser remanejados (e de que forma) para "encaixar" na agenda um evento importante?

Perguntas tão diversas como estas têm de ser respondidas a todo momento. Geralmente, devido à falta de ferramentas computacionais adequadas, estes problemas são resolvidos de forma "manual", isto é, através de contatos pessoais, ligações telefônicas, e extensas trocas de emails, sem a utilização de quaisquer técnicas que poderiam auxiliar a geração de agendamentos mais eficientes e de forma mais rápida.

1.1 Motivação

O desenvolvimento de sistemas de agendamento em grupo adequados para cenários mais complexos ainda é um problema aberto para o qual não há solução satisfatória. Se um grupo com $n \geq 3$ pessoas deseja marcar um evento, cada participante apresenta suas disponibilidades e preferências através de um ou mais destes mecanismos "manuais", e tenta-se chegar a um acordo sobre um agendamento aceitável. Muitas vezes, por questões de organização, uma pessoa fica a cargo da coordenação desta troca de informações, mas, à medida em que o valor de n cresce, mesmo esta centralização não garante que um período

de tempo livre comum a todos os participantes será encontrado. Ainda que o caráter do evento admita que algumas das pessoas envolvidas em seu agendamento não estejam presentes, a própria tentativa de encontrar uma melhor solução na qual o maior número possível de pessoas possa participar torna-se dispendiosa.

A adoção de soluções proprietárias já mudou de alguma forma este cenário, quando todos os participantes do evento pertencem à mesma organização, mas, com a crescente complexidade das relações sociais, dificilmente este é o caso.

Embora estas aplicações de compartilhamento de calendários e agendamento tenham historicamente sido classificadas como sistemas PIM (*personal information management*), concordamos com Erickson [30], que argumenta que sistemas de informação para os quais a intensa troca de informações num contexto coletivo é a regra devam ser estudados sobre uma perspectiva mais ampla, a dos sistemas que ele chama de GIM (*group information management*).

Mais do que os aspectos de organização, busca e visualização de informações que são objeto de aplicações PIM, são inerentes a sistemas GIM o desenvolvimento de sofisticados mecanismos de compartilhamento e o estudo das implicações sociais da disponibilização de dados privados. Nas aplicações de agendamento, isto se traduz na necessidade de protocolos para suportar a troca de informações e as negociações de agendamento entre usuários de diversos sistemas, bem como mecanismos de segurança e acesso a estas informações.

Além disso, são demandadas funcionalidades para lidar com agendas com muitas restrições e que levam a situações nas quais o reagendamento de compromissos é necessário, ainda não presentes nas soluções de agendamento em grupo atuais.

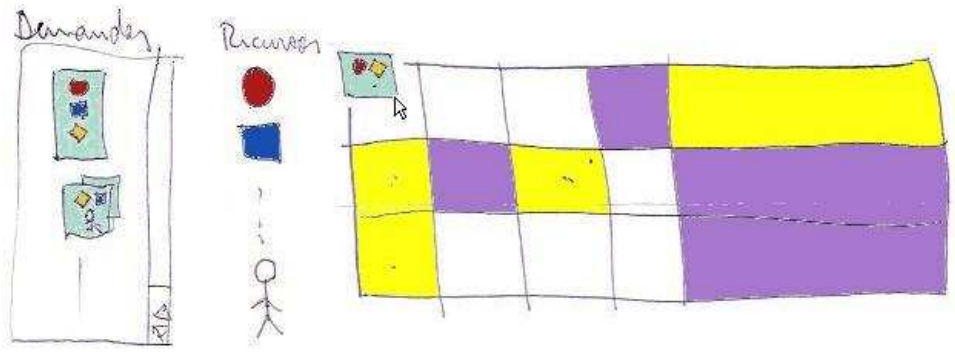


Figura 1.1: Esboço de uma ferramenta de agendamento mais inteligente

A idéia de uma ferramenta mais inteligente para o agendamento de compromissos vem sendo amadurecida desde 2004, quando foi desenvolvido o trabalho final de curso "Framework gráfico para um escalonador genérico de recursos" [73], no qual a necessidade de uma poderosa interface para uma aplicação como esta já era identificada. Como esboçamos na figura 1.1, tal sistema também deveria ser capaz de sugerir as melhores opções para o

(re-)agendamento de atividades, destacando visualmente os períodos de tempo mais adequados para tanto.

Desde então, percebemos a necessidade de compreender até que ponto os protocolos existentes seriam capazes de viabilizar a satisfação destas demandas prementes, e quais recursos de otimização e inteligência artificial poderiam ser utilizados para implementá-las. Isto indicou que, mais do que conceber uma aplicação com inovações, nosso trabalho deveria consistir em um estudo sistêmico sobre soluções para agendamento em grupo.

1.2 Objetivos

Este trabalho tem como objetivos: (1) fazer uma análise da evolução e do estado-da-arte alcançado pelas aplicações de agendamento em grupo e (2) propor uma abordagem inovadora – interdisciplinar – com base nesta análise que viabilize o surgimento de aplicações agendamento em grupo capazes de lidar de forma mais eficiente com cenários mais complexos, nos quais diversas pessoas tentam marcar um evento e, possivelmente, seja necessário reagendar compromissos.

Para tanto, detalhamos o histórico do desenvolvimento dos protocolos de agendamento, bem como a aplicação de promissoras técnicas dos campos de otimização, inteligência artificial e interfaces. Acreditamos que esta revisão bibliográfica oferece uma contribuição importante para a concepção de novas soluções, pois mesmo em língua inglesa inexiste uma análise articulada das influências mútuas entre essas áreas de conhecimento com relação a agendamento em grupo.

Com base nos sucessos e falhas das abordagens até hoje propostas, possíveis caminhos são apresentados em direção a uma nova geração de ferramentas de agendamento em grupo, mais inteligentes e poderosas, que irão aumentar a produtividade das atividades de agendamento ao minimizar os esforços de coordenação para administração do tempo. Para tanto, uma proposta de extensão dos protocolos padrão de agendamento é detalhada. Descrevemos, ainda, a arquitetura de uma moderna solução para agendamento em grupo, batizada de iScheduler, e relatamos as iniciativas de seu desenvolvimento[1] com respeito às extensões propostas e a um algoritmo de otimização para reagendamento.

Neste intuito, nenhuma área de conhecimento em particular é privilegiada. De fato, procuramos identificar a contribuição que elas podem oferecer para o desenvolvimento de melhores ferramentas de agendamento em grupo. Ao optar por uma abordagem em largura, certamente detalhes específicos relativos a cada uma delas são omitidos.

[1] No *technical report* IC-07-20 "New approaches for groupware scheduling" [75], tivemos oportunidade de apresentar preliminarmente, de forma mais sucinta, esta nova abordagem, com exceção aos resultados relativos a sua implementação prototipada.

1.3 Organização

No próximo capítulo apresentamos brevemente alguns conceitos, de diversas áreas de conhecimento, relevantes para este estudo.

A análise das abordagens para agendamento em grupo é feita no capítulo 3, onde, a partir de uma revisão cronológica, delineamos uma taxonomia para os sistemas de agendamento e examinamos os aspectos positivos e limitações das abordagens já propostas.

No capítulo 4, detalhamos a proposição de uma nova abordagem para agendamento em grupo, que indica caminhos para o desenvolvimento de ferramentas inovadoras e inclui o projeto de extensões de protocolos de agendamento para viabilizar novas funcionalidades.

O capítulo 5 traz a descrição conceitual do iScheduler, e reporta as experiências na implementação das extensões propostas no capítulo anterior, bem como de um algoritmo de otimização interativo para reagendamento que faz parte de sua arquitetura.

O último capítulo desta dissertação é dedicado à exposição de algumas considerações que devem ser objeto de reflexão para futuros projetos de pesquisa e desenvolvimento na área.

A especificação das extensões que propomos para os protocolos padrão de agendamento e o detalhamento técnico das implementações realizadas são apresentados, respectivamente, nos apêndices A e B.

Capítulo 2

Conceitos

Podemos definir abstrata e genericamente o problema do agendamento de recursos como um conjunto de recursos e suas disponibilidades, um conjunto de atividades e suas demandas por recursos, e um conjunto de restrições que devem ser atendidas, com o objetivo de agendar de forma ótima as atividades e os recursos ao longo do tempo.

Tipicamente, como uma conseqüência de suas origens industriais, problemas de agendamento são representados através da notação *job-machine*, e expressos como [62]:

- um conjunto de m tarefas (*jobs*), que representam as atividades; e
- um conjunto de n máquinas (*machines*) para as quais as tarefas devem ser agendadas. Estas máquinas podem representar quaisquer tipos de recursos, como equipamentos, salas, e o tempo das pessoas.

Diversos objetivos podem ser estabelecidos para este agendamento, como maximizar o número de tarefas completadas dentro de um certo limite de tempo. Por exemplo, problemas de agendamento cujo objetivo é maximizar ou minimizar alguma função nos tempos de conclusão das tarefas são genericamente chamados de problemas *jobshop*. Uma possível função objetivo para jobshop é a minimização do *makespan*, que consiste na duração total de um agendamento, determinada pelo término da última tarefa.

A partir destas definições, examinamos, a seguir, alguns conceitos de diversas disciplinas envolvidas, mesmo que tangencialmente, no estudo de soluções para agendamento em grupo. Além de fornecer subsídios para as discussões realizadas ao longo deste trabalho, a variedade de áreas de conhecimento abordadas nas próximas seções já pretende ilustrar a necessidade de uma abordagem interdisciplinar para o problema.

2.1 Otimização combinatória e Inteligência artificial

Muitos dos problemas de agendamento são NP-difíceis [33]. Não são conhecidos algoritmos polinomiais exatos para tratar esta classe de problemas, cujo número de soluções possíveis cresce exponencialmente em função do tamanho da entrada. Não é possível garantir a otimalidade de uma solução senão por uma busca exaustiva sobre as possibilidades, o que se inviabiliza à medida em que as instâncias consideradas aumentam.

As alternativas para lidar com problemas NP-difíceis podem ou não abrir mão da otimalidade desejada [6]. Técnicas enumerativas como programação dinâmica, *branch and bound*, *branch and cut* e *branch and price* introduzem pequenas modificações em algoritmos exponenciais na esperança de que, na prática, estes caminhem mais rapidamente para a solução ótima e os tempos de execução sejam razoáveis para a maioria das instâncias.

Outros métodos, como heurísticas, busca local e algoritmos de aproximação, geram algoritmos polinomiais que exploram características particulares do problema a ser resolvido e podem resultar em soluções não muito distantes da ótima, mas que de modo geral são boas o suficiente para a aplicação desejada. Em particular, os algoritmos de aproximação [83] fazem uso de técnicas formais para demonstrar que a distância entre a solução obtida e a solução ótima nunca será maior do que um determinado fator de aproximação.

Problemas de otimização estão inseridos em uma classe geral de problemas que podem ser resolvidos por meio de busca [66].

Basicamente, um problema de busca caracteriza-se por um estado inicial e uma função sucessor que gera os próximos estados possíveis. Uma solução ótima consiste em um estado objetivo que deve ser alcançado pelo processo de busca. Algoritmos de busca geram um grafo de busca resultante do percurso entre os estados e seus sucessores, no qual os nós correspondem aos estados e as arestas correspondem à expansão entre estados antecessores e sucessores. Um custo de caminho, que reflete a medida de desempenho do percurso entre dois nós, pode ser atribuído a cada aresta deste grafo, e um fator de ramificação b indica o número máximo de sucessores de qualquer nó.

Em problemas de agendamento, o estado inicial é dado pelo agendamento atual, a função sucessor é composta pelas operações de agendamento possíveis (e.g., associar uma tarefa a uma máquina, ou mudar a seqüência de execução de uma tarefa na máquina), e o universo de estados possíveis é formado por todos os agendamentos que podem ser gerados pela aplicação das operações de agendamento.

Os algoritmos de busca são qualificados conforme sua capacidade de encontrar uma solução ótima, se ela existir, e pelos seus requisitos de espaço e tempo de execução. O que determina a qualidade de um algoritmo é a sua estratégia de busca, que consiste na política adotada para gerar os próximos nós a serem percorridos no grafo de busca, a partir do nó raiz que corresponde ao estado inicial.

Algumas estratégias não exigem qualquer conhecimento específico sobre o problema

em questão, e são chamadas de estratégias de busca sem informação.

Na busca em extensão, ou em largura, todos os sucessores do nó raiz são expandidos, depois os sucessores desses nós, e assim por diante. Se o nó objetivo mais raso estiver em alguma profundidade finita d no grafo de busca, a busca em extensão encontra a solução ótima. Contudo, por ter complexidade de espaço (e de tempo) $O(b^{d+1})$, já que todos os nós com sucessores inexplorados devem ser armazenados, ela se torna inviável para a maioria dos problemas práticos.

A busca em profundidade sempre expande o nó mais profundo na borda atual do grafo de busca, prosseguindo até suas folhas, onde os nós não têm sucessores. Conforme esses nós são expandidos, a busca retorna ao nó seguinte mais raso com sucessores inexplorados. Como apenas um único caminho da raiz até o nó folha precisa ser armazenado, já que um nó pode ser removido da memória após todos os seus sucessores serem explorados, sua complexidade de espaço para uma profundidade máxima m é $O(bm)$. Sua complexidade de tempo, entretanto, é $O(b^m)$, o que também impossibilita sua aplicação em muitas situações.

A busca em profundidade limitada considera que todos os nós na profundidade l não têm sucessores, já que a busca em profundidade será inconclusiva se $m = \infty$. Contudo, ela é incompleta, isto é, não há garantia de que o objetivo será alcançado, pois caso o nó com a solução procurada esteja em uma profundidade $d > l$, ele não será expandido.

A busca por aprofundamento iterativo é uma estratégia que combina os benefícios da busca em profundidade e da busca em extensão. Ela consiste em executar diversas vezes a busca em profundidade limitada aumentando gradualmente o limite l – 0, 1, 2, e assim por diante – até encontrar um objetivo. Ao contrário da busca em profundidade limitada, ela é completa, pois expande o nó da solução quando seu limite de profundidade alcança d, tendo complexidade de tempo $O(b^d)$. O custo de gerar os nós mais rasos por diversas vezes não chega a ser um problema, pois se o fator de ramificação b for aproximadamente o mesmo para todos os níveis, a maioria dos nós tenderá a estar em profundidade maior do que d, e não chegará a ser expandida. Por ter complexidade de espaço $O(bd)$, a busca por aprofundamento iterativo é em geral o método de busca sem informação mais recomendado para grandes espaços de busca cuja profundidade da solução ótima é desconhecida.

Para problemas reais, entretanto, as estratégias de busca sem informação são ineficientes. Estratégias de busca com informação utilizam conhecimento específico do problema para encontrar soluções de forma mais eficiente.

Uma abordagem geral para busca com informação é a busca pela melhor escolha. Seja $f(n)$ a função de avaliação de um nó n que estima a distância de n até o objetivo, advinda do conhecimento específico do problema. A busca pela melhor escolha consiste em sempre expandir o nó com o menor valor de $f(n)$.

Chama-se de busca gulosa a estratégia de busca pela melhor escolha que consiste em aplicar em um nó n uma função heurística $h(n)$ que estima o custo do caminho mais

econômico de n até um nó objetivo e fazer $f(n) = h(n)$. A cada iteração, a busca gulosa avança em profundidade no grafo de busca, não fazendo *backtracking*. Embora tenha complexidades de tempo e espaço iguais à da busca em profundidade, no pior caso, uma boa função heurística pode fazer com que, na prática, sua complexidade tenha uma redução substancial.

A forma aprimorada e mais conhecida da busca pela melhor escolha é a busca A*. Ela avalia, de forma combinada, o custo $g(n)$ do caminho desde o nó raiz até n, e o custo estimado para alcançar o objetivo, fazendo $f(n) = g(n) + h(n)$. Assim, $f(n)$ torna-se custo estimado da solução mais econômica passando por n. Se a função heurística $h(n)$ for admissível, isto é, nunca superestimar o custo para alcançar o objetivo, a busca A* garante que a solução ótima será encontrada, pois o nó objetivo será expandido. Quanto melhor a estimativa dada pela função $h(n)$, melhor o desempenho da busca.

As heurísticas para algoritmos de busca com informação podem ainda fazer uso de estratégias de aprendizagem específicas, a fim de aprimorar as estimativas da função de avaliação, fazendo com que nós que recorrentemente não levam à solução deixem de ser expandidos e aumentando a eficiência da busca.

Algoritmos de busca local são um caso particular de busca pela melhor escolha. Eles são especialmente úteis para problemas de otimização, nos quais apenas a solução importa e o caminho da raiz até o nó objetivo é irrelevante. Algoritmos de busca local não armazenam o caminho até o objetivo, mantendo na memória apenas o estado corrente. A cada iteração, os estados vizinhos são considerados para expansão.

Como é possível que o algoritmo entre em um ciclo de repetição de nós já visitados, devido ao fato de o caminho desde o nó raiz não ser armazenado, uma estratégia de busca tabu pode ser adotada. Ela consiste em manter uma lista de movimentos tabu, que contém estados já expandidos em um passado próximo e que não devem ser visitados novamente.

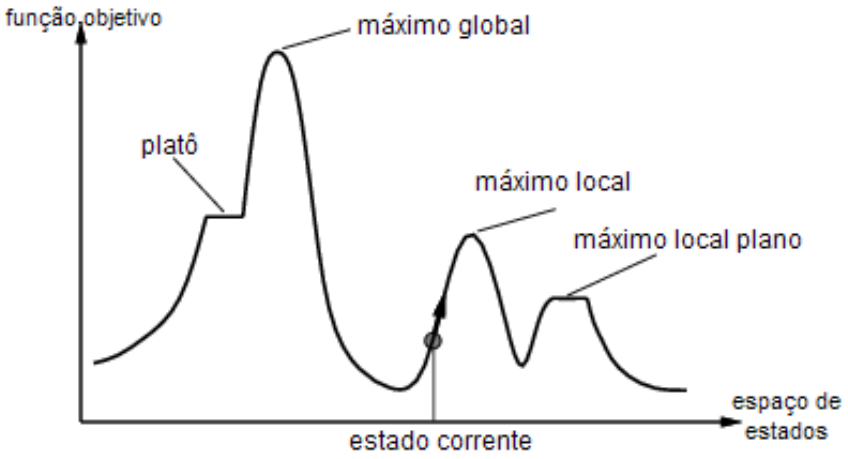


Figura 2.1: Topologia de espaço de estados (Adaptado de [66])

A figura 2.1, que traz uma topologia de espaço de estados para um problema de maximização, ilustra as dificuldades que podem envolver a adoção de uma estratégia de busca local gulosa: o processo de busca pode ficar paralisado por um máximo local, pois nenhuma solução vizinha será melhor do que a corrente.

Estratégias de busca bastante utilizadas atualmente têm mecanismos para mitigar este problema. A busca de têmpera simulada (*simmulated annealing*) executa movimentos aleatórios – cuja freqüência decresce ao longo do processo – que momentaneamente podem piorar a solução, mas são capazes de deslocar a busca de um pico local. Algoritmos genéticos mantém uma população dos k melhores estados obtidos, gerando os sucessores através de uma combinação de dois estados pai e de operações de mutação que, com uma pequena probabilidade, aplicam modificações aleatórias aos cruzamentos que podem gerar soluções melhores.

2.2 Redes e protocolos

Redes de computadores são organizadas através de pilhas de protocolos cujas camadas oferecem serviços específicos determinados por uma interface de comunicação [82]. Este desacoplamento garante que, mantidas as interfaces, o protocolo de uma camada possa ser substituído por outro sem perda de generalidade.

O modelo de referência de redes Open Systems Interconnection (OSI) define 7 camadas de protocolos:

1. Física (*physical*) – determina as especificações físicas para os dispositivos de comunicação, e os padrões elétricos, magnéticos ou ópticos utilizados para transmissão de dados.

2. Enlace de dados (*data link*) – provê diretivas para a transferência de dados entre os componentes de uma rede e controle de acesso ao meio físico compartilhado, além de detectar e corrigir erros do meio físico.

3. Rede (*network*) – controla o endereçamento de pacotes de dados, a fim de que possam transitar da origem ao destino através das rotas de comunicação disponíveis.

4. Transporte (*transport*) – provê uma transferência transparente de dados, formando pacotes a serem transmitidos pela camada de rede e garantindo que, no destino, eles possam ser organizados na mesma ordem em que foram enviados.

5. Sessão (*session*) – permite que diferentes computadores estabeleçam uma sessão de comunicação, iniciando, controlando e terminando conexões.

6. Apresentação (*presentation*) – estabelece um contexto entre os dados transmitidos pela camada de sessão e os dados a serem recebidos pela camada de aplicação, ficando a cargo de tarefas como criptografia e compressão de dados.

7. Aplicação (*application*) – faz a interface direta com as aplicações, definindo serviços de alto nível como transferência de arquivos e envio de emails.

Embora seja uma referência conceitual para redes de computadores, o modelo OSI não é utilizado de fato. As camadas de protocolos que organizam as comunicações na Internet são representadas por dois de seus mais conhecidos protocolos, o Transmission Control Protocol (TCP) e o Internet Protocol (IP), fazendo com que este modelo prático seja mais conhecido como modelo TCP/IP.

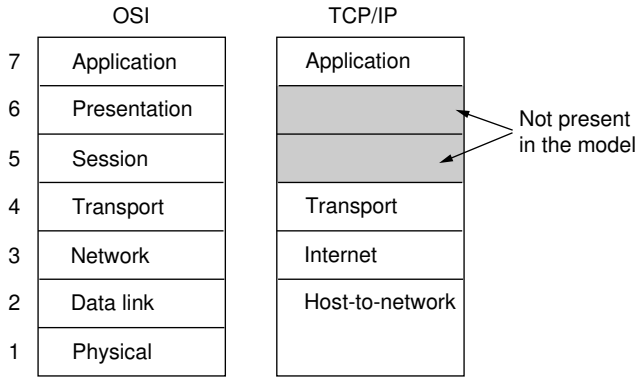


Figura 2.2: Modelos de referência OSI e TCP/IP [82]

A figura 2.2 ilustra a correspondência entre as camadas dos modelos OSI e TCP/IP. No modelo TCP/IP, não há uma separação clara entre as camadas física e de enlace de dados. Padrões como Ethernet já contêm, simultaneamente, especificações para o meio físico e componentes de protocolo para controle de acesso e correção de erros. Além disso, as camadas de sessão e apresentação não são implementadas separadamente, mas em conjunto com as funcionalidades da camada de aplicação.

Devido ao êxito do HyperText Transfer Protocol (HTTP) na implementação da Web, o conjunto dos protocolos mais comuns da Internet, ilustrados na figura 2.3, também é conhecido como pilha HTTP.

2.2.1 O desenvolvimento de padrões e a IETF

Diversas organizações têm desempenhado um papel relevante no desenvolvimento de protocolos para a Internet, como o Institute of Electrical and Electronics Engineers (IEEE),

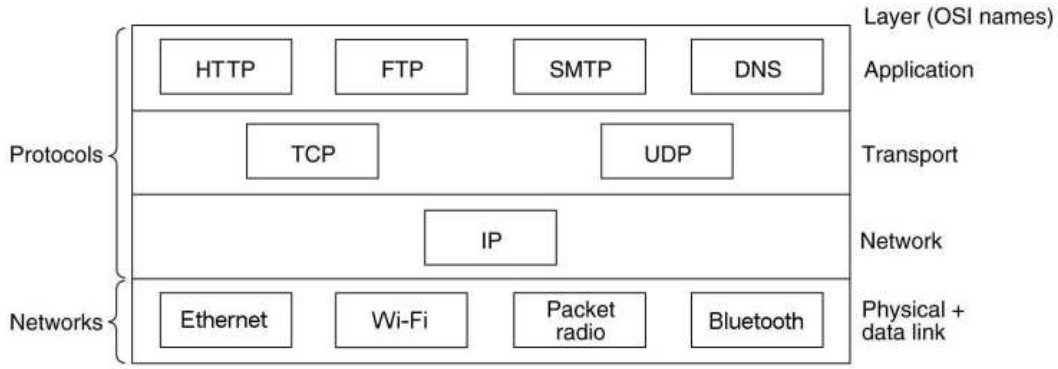


Figura 2.3: Alguns dos protocolos mais comuns da Internet (Adaptado de [82])

responsável pelo padrão de comunicação 802.11 (Wi-Fi) para redes sem fio; a International Standards Organization (ISO), que desenvolveu o modelo de referência OSI; e o World Wide Web Consortium (W3C), que promove padrões como HTML e XML.

Dentre estes organismos, destaca-se a Internet Engineering Task Force (IETF), uma associação dedicada especificamente ao desenvolvimento de padrões para a Internet. A IETF é organizada em diversos grupos de trabalho que trabalham com diferentes ênfases. Enquanto alguns estão mais voltados para protocolos da camada de aplicação, outros são focados em padrões de comunicação no nível *host-to-network*. Importantes protocolos, como IP e HTTP, foram desenvolvidos através da IETF[1].

O desenvolvimento de padrões na IETF segue um processo extremamente organizado no qual drafts de especificações são submetidos para debate e aprimoramento, em listas de discussão e nos encontros presenciais que ocorrem três vezes ao ano. Havendo consenso de que um draft tem interesse coletivo e atingiu maturidade suficiente, ele pode ser publicado como uma Request For Comments (RFC).

As RFCs constituem uma série numerada de documentos permanentes que descrevem padrões de protocolos e procedimentos. Por exemplo, o HTTP é especificado na RFC 2616 [32], e o próprio processo de desenvolvimento de padrões é descrito numa RFC, a 2026 [9]. RFCs são documentos não modificáveis que só podem evoluir através da publicação de uma nova RFC.

2.3 Sistemas distribuídos

Sistemas distribuídos são um caso particular no qual uma rede de computadores independentes age como um único e coerente sistema [82]. Este comportamento é garantido por uma camada de software que coordena suas ações através da troca de mensagens, a fim

[1]Uma lista de importantes protocolos da Internet publicados como RFCs pode ser encontrada em `http://www.rfc-editor.org/rfcxx00.html`.

de garantir um alto grau de coesão e transparência [13].

Um dos problemas centrais de algoritmos distribuídos é a falta de um relógio global: não há uma referência de tempo absoluta com a qual os computadores de um sistema distribuído possam se sincronizar, o que inviabiliza garantias a respeito do estado global do sistema (e.g., a seqüência temporal de eventos de trocas de mensagens). O conceito de relógios lógicos, introduzido por Lamport [46], oferece uma solução para este problema:

1. Um contador L_i é incrementado a cada evento ocorrido em um processo p_i;
2. Quando p_i envia uma mensagem m, envia também o valor L_i;
3. Ao receber (m, L_i), um processo p_j computa $L_j = max(L_j, L_i)$ e aplica 1.

Para dois eventos x e y, denotamos $x \rightarrow y$ se x precede y e $x \mid\mid y$ se x ocorre paralelamente a y. Com a utilização de relógios lógicos, temos que $x \rightarrow y \Rightarrow L(x) < L(y)$. A figura 2.4 mostra, para 3 processos p_1, p_2 e p_3, a evolução de seus relógios lógicos à medida em que os eventos ocorrem.

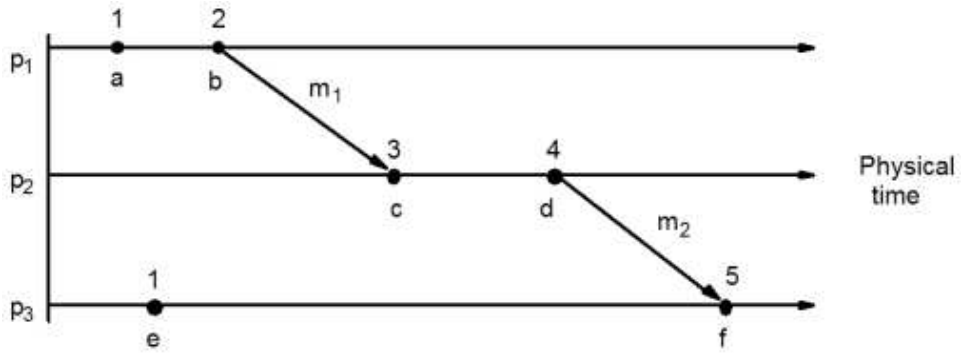


Figura 2.4: Temporização de eventos com relógios lógicos [13]

Note que este mecanismo não garante que $x \rightarrow y$ se $L(x) < L(y)$. Por exemplo, $L(b) > L(e)$, mas $b \mid\mid e$.

O uso de relógios lógicos é um mecanismo fundamental para viabilizar a coordenação entre os componentes de um sistema distribuído, por exemplo, quando estes devem chegar a um consenso sobre o valor de uma variável após um ou mais processos sugerirem valores para ela.

2.4 Agendamento em grupo

Agendamento em grupo é um processo de tomada de decisão inerentemente distribuído, no qual os atributos de um evento são as variáveis sobre as quais deve-se chegar a um

consenso. Vale ressaltar a clara distinção existente entre atividades de compartilhamento de calendários[2] (*calendaring*), que não têm "efeitos colaterais" na organização do tempo, e atividades de agendamento (*scheduling*).

2.4.1 Formalizando agendamento distribuído

Deve-se a Sen e Durfee [71] o mais conhecido estudo formal sobre agendamento distribuído, feito com base em um protocolo de negociação genérico que descreve como participantes de um evento[3] em grupo convergem para um horário durante o processo de negociação:

1. O organizador[4] envia uma proposta de horário para o evento.
2. Os participantes[5] respondem para o organizador.
3. Se não há consenso na aceitação da proposta do organizador, volta-se para o passo 1, ou interrompe-se a negociação caso o número máximo de iterações tenha sido atingido.

A eficiência de uma negociação pode ser medida em termos do número de iterações (ou *rounds*) necessárias para se atingir o consenso, e no tamanho das mensagens trocadas entre o organizador e os participantes.

Três tipos de estratégias de negociação envolvendo este protocolo foram estudadas:

- as estratégias de anúncio (*announcement strategies*) definem, no passo 1 da negociação, se o organizador oferece somente uma opção de horário – estratégia única (*best*) – ou n opções de horário – estratégia múltipla (*good*);
- as estratégias de resposta (*bidding strategies*) definem, no passo 2 da negociação, se os participantes simplesmente aceitam ou rejeitam a proposta do organizador – estratégia sim-não (*yes-no*) – ou fazem uma contra-proposta – estratégia com alternativas (*alternatives*); e
- as estratégias de comprometimento (*commitment strategies*) definem, quando um organizador ou participante estão envolvidos em diversas negociações simultâneas, se os horários sugeridos por outros são provisoriamente reservados em suas agendas, de forma que nenhum outro evento possa ser agendado nestes horários – estratégia comprometida (*committed*) – ou não – estratégia não-comprometida (*non-committed*).

[2] Neste contexto, utilizamos o termo "calendários" como sinônimo de "agendas".

[3] Neste contexto, utilizamos o termo "evento" como uma referência genérica para qualquer entidade agendável, como atividades, tarefas e compromissos [50].

[4] *Organizer*, *host* ou *initiator*.

[5] *Attendees*, *invitees* ou *participants*.

As duas primeiras estratégias são relativas a comunicação. Elas equilibram a necessidade de privacidade (que implica em trocar menos informações) com demandas por rápida convergência no consenso sobre os horários de eventos (que pode ser agilizada com a troca de mais informações).

Sen e Durfee mostram que as estratégias de anúncio múltiplo e de resposta com alternativas levam a uma convergência mais rápida na negociação de agendamento, especialmente quando as agendas dos participantes são densas e a duração dos eventos é considerável. Em outras palavras, na medida em que as pessoas estão dispostas a compartilhar mais informações sobre suas agendas, os sistemas de agendamento em grupo podem tirar proveito disto para tornar o processo de negociação mais eficiente.

Acreditamos que, embora ambas as estratégias incorram em um custo de comunicação maior, dado que as mensagens carregam mais informação, o incremento no tamanho das mensagens não é significante para representar uma sobrecarga na rede. Quando os recursos de comunicação não são um gargalo, o que é verdade para as situações típicas, o número de rounds se torna a única medida representativa da eficiência de uma negociação.

Por fim, as estratégias de comprometimento são relevantes no caso de múltiplas negociações concorrentes de agendamento. Duas ou mais negociações em curso podem eventualmente competir pelo mesmo período de tempo até que os horários dos eventos sejam confirmados. Quando a estratégia comprometida é adotada, ela torna as agendas mais densas ao reservar recursos. Isto reduz as opções de horários para outras negociações, tornando-as potencialmente mais extensas e difíceis, mas também evita que um horário em consideração seja ocupado por outro evento antes do encerramento de sua negociação.

Estas interdependências entre negociações devem ser apropriadamente tratadas, para evitar, ao mesmo tempo, que eventos cujas negociações são mais extensas não encontrem um período de tempo disponível ao final do processo, e que períodos de tempo não sejam inutilmente reservados, dificultando artificialmente outras negociações de agendamento concomitantes e levanto a resultados sub-ótimos. O uso das estratégias de comprometimento pode ser dosado, por exemplo, utilizando a estratégia comprometida para eventos de alta prioridade, e a estratégia não-comprometida para os demais.

2.4.2 Reagendamento

Em cenários muito restritos não há, contudo, outra opção senão desmarcar eventos. Cancelar um evento em favor de outro é uma decisão importante, porque um reagendamento pode ainda implicar em perturbações em cascata ao longo de uma cadeia de usuários conectados por eventos em comum.

O problema de quando reagendar um evento foi investigado por Modi e Veloso [53], que identificaram que o uso de estratégias de reagendamento previamente fixadas, seja sempre ou nunca reagendar um dado evento para abrir espaço na agenda a um novo evento,

leva a agendamentos sub-ótimos ou infactibilidades, mostrando-se inadequadas. Logo, da mesma forma que a estratégia de comprometimento, a estratégia de reagendamento deve ser adotada em função de situações específicas.

Vemos o reagendamento de eventos como parte do fluxo normal de agendamentos em grupo, e não como uma exceção. Mudanças imprevistas certamente ocorrerão: elas devem ser esperadas e melhor suportadas pelos sistemas de agendamento em grupo. Caso contrário os usuários são, novamente, compelidos a usar mecanismos externos e dispendiosos, como telefone e email, para resolver estas situações.

2.5 Interfaces

Uma interface adequada à execução do trabalho do usuário é tão fundamental para o sucesso de uma aplicação quanto o conjunto de funcionalidades disponibilizadas [63].

A aceitabilidade é a combinação de todos os fatores considerados pelo usuário ao avaliar a qualidade de um software, tais como confiabilidade, custo e compatibilidade (ou interoperabilidade). Estes critérios certamente são considerados, mas indubitavelmente a usabilidade é o fator de aceitabilidade que determina de forma decisiva se um sistema será utilizado ou não. O conceito de usabilidade engloba atributos como adequação à tarefa desempenhada, facilidade de aprendizado, eficiência e satisfação subjetiva no uso do sistema [2].

Com a finalidade de aumentar o nível de usabilidade das interfaces, cada vez mais a noção de metáforas é adotada na concepção de sistemas. Metáforas – como "pasta", "lixeira", "recortar" e "colar" – são termos resignificados que utilizam a experiência pessoal do usuário para definir uma entidade computacional com a qual ele não tem familiaridade em termos de outra, sem a necessidade de utilizar o jargão técnico que dificultaria essa tarefa.

O uso de metáforas foi especialmente enfatizado pelas interfaces de manipulação direta, emergentes a partir da década de 80. Estas interfaces possuem recursos gráficos – como botões, ícones, menus e *drag and drop* – que proporcionam ao usuário a "ilusão" de que os objetos do sistema são manipulados diretamente, eliminando a necessidade de comandos e sintaxes especiais a serem aprendidos. O uso de recursos gráficos para representar abstrações do domínio da tarefa é uma tendência para minimizar o esforço cognitivo do usuário na operação dos sistemas.

Capítulo 3

Revisão das abordagens para agendamento em grupo

A utilização de computadores para realizar atividades de agendamento não é uma idéia recente [69]. Nos anos 60 e 70, com a descoberta de importantes resultados de teoria de complexidade, e a constatação de que a maioria dos problemas de agendamento eram NP-difíceis, as iniciativas de pesquisa concentraram-se no projeto de algoritmos que resolvessem mais eficientemente estes problemas.

As primeiras tentativas de computadorizar rotinas de agendamento foram feitas com foco em áreas de planejamento bem específicas, como agendamento de aeronaves [10] e escalonamento de disciplinas, professores e salas de aula em universidades [36].

Na década de 80, com a popularização dos microcomputadores nos escritórios, os primeiros estudos sobre o uso de sistemas de calendários neste ambiente foram conduzidos. Naquele momento, a crescente preocupação com problemas de usabilidade motivaram a análise de fatores psicológicos relacionados com a informatização das atividades de agendamento.

Kelley e Chapanis [41] mostraram que as preferências individuais no uso de calendários pessoais seriam determinantes para a adoção de sistemas computadorizados de calendários e agendamento, pois as pessoas têm modos particulares de utilizar as agendas em papel.

Com base nesta constatação, Kincaid et al. [42] realizaram o levantamento de uma série de requisitos obrigatórios e desejáveis que estes sistemas deveriam satisfazer. Demandas por flexibilidade na administração de calendários, com o uso de interfaces gráficas, múltiplas formas de visualização (diária, semanal, mensal), lembretes de compromissos, reserva de recursos e agendamento automático, entre outras facilidades, influenciaram fortemente o desenvolvimento de aplicações desde então, e se tornaram a base para as ferramentas contemporâneas.

Diversos "automatizadores" de agendamento para escritórios foram então propostos. Greif e Sarin [35], por exemplo, apresentaram os protótipos MPCAL e RTCAL, sistemas

de agendamento que tinham como premissas um ambiente centralizado e a existência de um banco de dados compartilhado. Este banco de dados coordenaria todas as etapas de agendamento, desde a busca por horários livres até a troca de propostas durante a negociação de eventos.

3.1 A primeira geração de sistemas de agendamento

Somente no final da década de 80, com a introdução daquele conhecido como o primeiro sistema PIM no mercado, o Lotus Agenda, a idéia de utilizar o computador como um organizador de tempo foi trazida para os usuários de computadores fora das universidades e empresas de tecnologia. O Lotus Agenda era uma aplicação DOS que permitia aos usuários inserir notas e organizá-las temporalmente.

No início dos anos 90, a Lotus substituiu o Agenda pelo Lotus Organizer, com uma inovadora interface gráfica baseada nas agendas de papel. Em seguida, a PeopleCube lançou o Meeting Maker para a plataforma Macintosh, e a Microsoft desenvolveu o Schedule+, uma aplicação Windows. Estes sistemas apresentavam interfaces de manipulação direta mais poderosas, utilizando metáforas baseadas em calendários reais, e marcaram a popularização inicial das aplicações de calendários.

Mais tarde, o Schedule+ foi substituído pelo Microsoft Outlook, um organizador integrado que incorporou suas funcionalidades às de email, agenda de contatos e anotações, dentre outras. Gradualmente, o Outlook tomou a liderança de mercado então exercida pelo Organizer, o qual posteriormente também evoluiu para um conjunto unificado de ferramentas chamado Lotus Notes.

Na medida em que o uso destas aplicações em ambientes corporativos se tornou comum, funcionalidades de compartilhamento de calendários e agendamento em grupo foram sendo incorporadas a elas, tornando possível a transição entre a administração de tempo particular e em grupo.

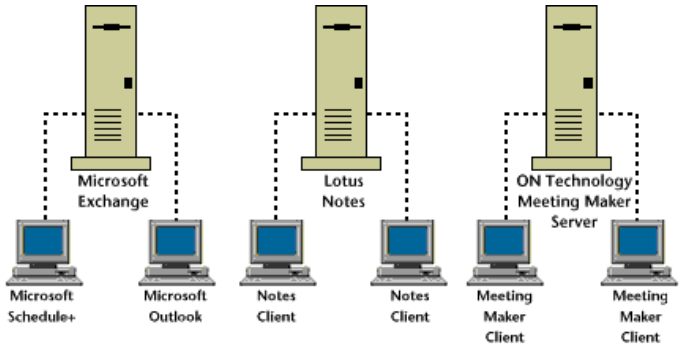


Figura 3.1: Compartilhamento de calendários e agendamento em grupo via soluções proprietárias [58]

Conforme mostrado na figura 3.1, a integração destes sistemas em um ambiente colaborativo foi feita através de servidores específicos que implementavam protocolos de comunicação proprietários. Microsoft Schedule+ e Outlook, por exemplo, somente poderiam conversar entre si, e com o uso do Microsoft Exchange Server.

Embora o compartilhamento de informações entre diferentes plataformas não fosse possível devido à falta de protocolos de comunicação, esta primeira geração de sistemas teve o mérito de automatizar tarefas rotineiras de administração do tempo e aumentar a produtividade dentro das organizações.

3.2 O estabelecimento de protocolos de comunicação[1]

A necessidade de tornar possível a interoperabilidade entre sistemas de agendamento distintos motivou, em 1996, a criação do IETF Calendaring and Scheduling Working Group (CalSch), dedicado ao estabelecimento de padrões abertos para trocas de informações de calendários e agendamento na Internet [57].

A figura 3.2 ilustra o cenário almejado por esta iniciativa: a comunicação entre um grande número de aplicações de calendário e agendamento possibilitada por padrões garantidores de interoperabilidade.

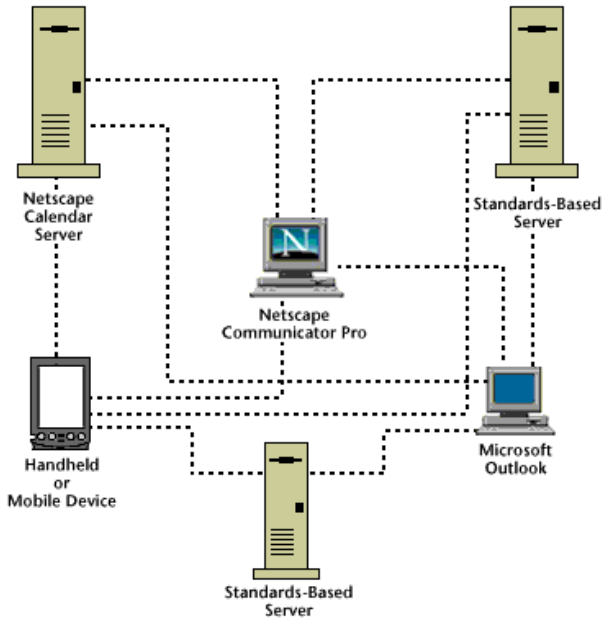


Figura 3.2: A integração de sistemas heterogêneos prevista pelo grupo CalSch [58]

[1]Uma detalhada revisão da história da interoperabilidade de sistemas de calendários e agendamento é apresentada no artigo de Dusseault e Whitehead [26], no qual partes desta e das próximas seções são baseadas.

Com base no trabalho prévio do Versit Consortium[2] no formato vCalendar [40] para dados de calendários e agendamento, o grupo CalSch desenvolveu e publicou em 1998 as seguintes especificações[3]:

- A RFC 2445 [22] – Internet Calendaring and Scheduling Core Object Specification (iCalendar[4]) – estabelece um modelo de dados que determina como representar textualmente informações de calendários e mensagens de agendamento.
- A RFC 2446 [76] – iCalendar Transport-Independent Interoperability Protocol (iTIP) – descreve um padrão para executar operações de manipulação de calendários e agendamento com objetos iCalendar.
- A RFC 2447 [21] – iCalendar Message-Based Interopability Protocol (iMIP) – define uma implementação do protocolo iTIP utilizando email.

3.2.1 iCalendar

A especificação iCalendar [22] define quatro **componentes** principais: *events*, *to-dos*, *free-busy time* e *journal entries*. Para ser válido, um **objeto** iCalendar deve conter pelo menos um destes componentes. A extensão padrão para arquivos com dados iCalendar é .ics.

Todos os componentes de calendário são descritos com o uso de **propriedades** obrigatórias ou opcionais, que por sua vez podem ter **parâmetros**. Por exemplo, os componentes do tipo *event* têm uma propriedade `SUMMARY` obrigatória que representa textualmente seu propósito.

O formato iCalendar não é baseado em XML ou qualquer outro padrão de representação de dados. Por isso, a especificação define uma sintaxe própria para representação dos componentes de calendário, e diversas exigências técnicas não-triviais, como uma limitação de 75 octetos[5] por linha.

Todo componente iCalendar é limitado por tags `BEGIN:<nome>` e `END:<nome>`. Suas propriedades são expressas na forma `<nome>:<valor>`. Parâmetros das propriedades são separados por ponto-e-vírgula (`;`) e são descritos na forma `<nome>=<valor>` antes dos dois pontos (`:`) que separam o nome e o valor da propriedade.

O exemplo da figura 3.3 mostra um objeto iCalendar que contém um evento "Defesa de mestrado" (propriedade `SUMMARY`), com início às 14h do dia 08/02/2008 (propriedade

[2] O Versit Consortium também foi responsável pela introdução do formato vCard para cartões de visita eletrônicos, comumente anexados a emails e usados por aplicações de administração de contatos.

[3] A RFC 3283 [49] explicita o relacionamento entre estas especificações.

[4] Também conhecida como iCal.

[5] Um octeto equivale a 8 bits. Embora em padrões de codificação de caracteres como o US-ASCII cada octeto corresponda a um caractere; em alguns padrões, como o Unicode, são necessários mais de 8 bits para representar um caractere.

```
BEGIN:VCALENDAR
PRODID:-//Unicamp//iScheduler//EN
VERSION:2.0
BEGIN:VEVENT
UID:20080108T090000Z-00001@example.com
DTSTAMP:20080108T090000Z
DTSTART:20080208T140000Z
DURATION:PT1H30M
SUMMARY:Defesa de mestrado
END:VEVENT
END:VCALENDAR
```

Figura 3.3: Exemplo de objeto iCalendar

`DTSTART`), e duração de 1h30min (propriedade `DURATION`). Note ainda a presença das propriedades obrigatórias de versão e identificação do produto (respectivamente, `VERSION`, cujo valor fixo estabelecido pela RFC 2445 é `2.0`[6], e `PRODID`, para o qual foi utilizado um valor fantasia) no objeto. A propriedade `UID` permite uma identificação única do componente `VEVENT`, de fundamental importância na sua utilização com o protocolo iTIP, como detalhado mais adiante.

Um mecanismo conhecido como *X-tokens* permite que componentes, propriedades e parâmetros não especificados no padrão sejam utilizados no escopo de uma aplicação, exigindo que eles sejam da forma `X-<identificação_do_produto>-<nome>` para evitar incompatibilidades. Por exemplo, uma aplicação poderia definir uma propriedade `X-ISCHED-TESTE` para representar dados não previstos na especificação iCalendar. Este mecanismo oferece flexibilidade adicional, ao especificar "um padrão para se fazer coisas fora do padrão" [20] que não é oneroso, pois desobriga outras aplicações de armazenar ou manter o conteúdo de X-tokens em mensagens de agendamento subseqüentes.

O componente VEVENT

Eventos (*events*) são representados pelo componente `VEVENT`. Eles constituem a principal entidade de calendários, sendo utilizados para todos os tipos de compromissos agendáveis que ocupam tempo em uma agenda, como atividades, reuniões e aniversários.

As propriedades mais importantes de um evento são sua data de início (`DTSTART`) e término (`DTEND`) – ou duração (`DURATION`) –, seu organizador (`ORGANIZER`), seus participantes (`ATTENDEE`s) e seu local (`LOCATION`), mas diversas outras informações também podem ser especificadas.

[6] A versão 1.0 corresponde ao padrão vCalendar.

O padrão iCalendar ainda dispõe de um poderoso mecanismo que permite a descrição de eventos recorrentes, que se repitam com uma determinada freqüência, através de regras de recorrência expressas pela propriedade `RRULE`.

Os componentes `VEVENT` podem, ainda, conter um ou mais componentes `VALARM`, cujo propósito é descrever uma rotina de alarme para relembrar o usuário do evento.

```
BEGIN:VEVENT
UID:20071005T133225Z-00001@example.com
DTSTAMP:20071005T133225Z
DTSTART:20071219T170000Z
DTEND:20071219T180000Z
SUMMARY:Consulta
ORGANIZER;CN=Doutor:doutor@example.com
ATTENDEE;CN=Doutor:doutor@example.com
ATTENDEE;CN=Paciente:paciente@example.com
LOCATION:Rua do Sorriso, 32
RRULE:FREQ=WEEKLY;BYDAY=WE;COUNT=10
BEGIN:VALARM
TRIGGER:-PT30M
REPEAT:2
DURATION:PT15M
ACTION:DISPLAY
DESCRIPTION:Lembrete! Dentista.
END:VALARM
END:VEVENT
```

Figura 3.4: Exemplo de evento recorrente

O exemplo da figura 3.4 mostra um evento recorrente com 10 instâncias que ocorrem nas quartas-feiras às 17h, a partir de 19/12/2007. Note também o uso do parâmetro `CN` (*common name*), nas propriedades `ORGANIZER` e `ATTENDEE`.

O componente VTODO

*To-do*s são usados para representar tarefas pendentes e seus status, e correspondem ao componente `VTODO`. A propriedade mais relevante de um to-do é seu prazo final (`DUE`). To-dos também podem conter componentes `VALARM`.

O exemplo da figura 3.5 mostra um to-do. A propriedade `CATEGORIES` é utilizada para classificar a tarefa, e o valor 5 para `PRIORITY` indica que ela tem prioridade média, pois os valores permitidos vão de 1 a 9. O valor da propriedade `PERCENT-COMPLETE` denota que metade da tarefa já foi cumprida.

```
BEGIN:VTODO
UID:20080108T090000Z-00001@example.com
DTSTAMP:20080108T090000Z
SUMMARY:Declarar imposto de renda
DUE:20080430T110000Z
CATEGORIES:BUSINESS
PRIORITY:5
PERCENT-COMPLETE:50
END:VTODO
```

Figura 3.5: Exemplo de *to-do*

Alternativamente, um to-do pode conter o parâmetro `DURATION`, com sua duração estimada, desde que não concomitantemente à presença do parâmetro `DUE`.

O componente VFREEBUSY

O componente `VFREEBUSY` permite a representação de períodos de tempo livres e ocupados (*free-busy time*). Estes componentes são usados, por exemplo, para descrever períodos de tempo não agendados que podem ser usados para guiar propostas de agendamento antes da negociação de um evento.

```
BEGIN:VFREEBUSY
UID:20080325T122800Z-00001@example.com
DTSTAMP:20080325T122800Z
ATTENDEE;CN=B:mailto:B@example.com
DTSTART:20080328T000000Z
DTEND:20080330T235959Z
FREEBUSY;FBTYPE=FREE:20080328T150000Z/PT3H
FREEBUSY;FBTYPE=FREE:20080329T163000Z/PT1H30M
FREEBUSY;FBTYPE=FREE:20080330T080000Z/PT6H
END:VFREEBUSY
```

Figura 3.6: Exemplo de *free-busy time*

A figura 3.6 mostra um componente free-busy que descreve, entre os dias 28 e 30/03/08, os seguintes períodos de tempo livre de um usuário fictício "B" através das propriedades `FREEBUSY`: das 15h às 18h no dia 28, das 16h30min às 18h no dia 29, e das 8h às 14h no dia 30.

Caso o propósito do componente seja expressar períodos de tempo ocupados, valores como `BUSY-TENTATIVE`, `BUSY` e `BUSY-UNAVAILABLE` podem ser utilizados no parâmetro `FBTYPE` da propriedade `FREEBUSY`.

O componente VJOURNAL

Os componentes de calendário `VJOURNAL` (*journal entries*) são os menos utilizados, e não têm um papel relevante em processos de agendamento. Como to-dos, journal entries não ocupam espaço em uma agenda. Seu propósito único é organizar informações dependentes de tempo (e.g., notas de um diário).

```
BEGIN:VJOURNAL
UID:20080208T170000Z-00001@example.com
DTSTAMP:20080208T170000Z
SUMMARY:Minuta da reunião
DESCRIPTION:1. Abertura\n
  2. Discussões principais\n
  3. Pendências\n
  4. Conclusão
END:VJOURNAL
```

Figura 3.7: Exemplo de *journal entry*

O exemplo da figura 3.7 mostra um componente journal que sumariza os assuntos tratados em uma reunião. Na prática, journal entries são apenas parcialmente implementados na maioria dos sistemas que adotam o padrão iCalendar [11].

3.2.2 iTIP

O protocolo iTIP [76] descreve conceitualmente como utilizar objetos iCalendar para executar operações de compartilhamento de calendários e agendamento. Isto é feito em termos dos **métodos** apresentados na tabela 3.1, que também detalha em quais situações eles são utilizados pelo organizador e pelos participantes de uma negociação de agendamento.

Para tanto, a propriedade `METHOD` do objeto iCalendar é utilizada, e seu valor indica qual a operação a ser executada pela mensagem de agendamento. O padrão iTIP impõe restrições sobre os componentes, propriedades e parâmetros que podem ser utilizados com cada método, a fim de que os objetos iCalendar estejam conformes com sua finalidade.

Conforme mostra a figura 3.8, nenhum protocolo de transporte específico é definido para a troca de mensagens iTIP que executam estes métodos.

Método	**Finalidade**	**Executado por**
`PUBLISH`	Publicar unilateralmente um componente de calendário para um ou mais usuários	Organizador
`REQUEST`	Requisitar a participação em um evento ou to-do; requisitar informações sobre *free-busy time*	Organizador, na maioria dos casos, e participantes, para delegar sua participação em um evento
`REPLY`	Responder a uma requisição, aceitando-a ou rejeitando-a	Participantes
`ADD`	Adicionar uma ou mais instâncias a um `VEVENT`, `VTODO` ou `VJOURNAL` existente	Organizador
`CANCEL`	Cancelar uma ou mais instâncias de um `VEVENT`, `VTODO` ou `VJOURNAL` existente	Organizador
`COUNTER`	Fazer uma contra-proposta para as propriedades de um componente de calendário	Participantes
`DECLINECOUNTER`	Rejeitar uma contra-proposta feita por um participante	Organizador
`REFRESH`	Requisitar a versão mais atual de um componente de calendário	Participantes

Tabela 3.1: Métodos iTIP

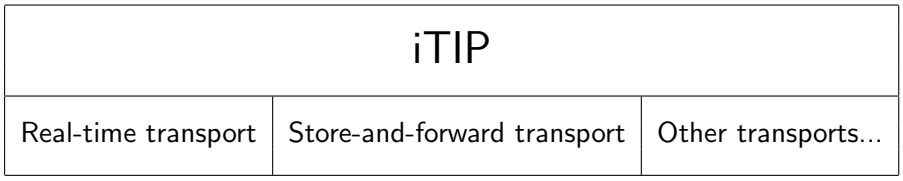


Figura 3.8: Arquitetura iTIP (Adaptado de [76])

A tabela 3.2 sumariza as combinações de componentes de calendário e métodos suportadas pela RFC 2446.

	VEVENT	VTODO	VJOURNAL	VFREEBUSY
PUBLISH	•	•	•	•
REQUEST	•	•		•
REPLY	•	•		•
ADD	•	•	•	
CANCEL	•	•	•	
COUNTER	•	•		
DECLINECOUNTER	•	•		
REFRESH	•	•		

Tabela 3.2: Componentes para os quais os métodos iTIP são válidos (Adaptado de [76])

A semântica destes métodos é descrita a seguir, com ênfase em sua atuação sobre componentes de eventos.

PUBLISH

O método `PUBLISH` é utilizado para publicar um evento, anunciando sua existência, e também para a publicação de to-dos, journal entries e free-busy time. Este é o método padrão para componentes iCalendar que não contenham a propriedade `METHOD`.

```
BEGIN:VCALENDAR
PRODID:-//Unicamp/iScheduler//EN
METHOD:PUBLISH
VERSION:2.0
BEGIN:VEVENT
UID:20071005T133225Z-00001@example.com
DTSTAMP:20071014T225108Z
ORGANIZER:mailto:A@example.com
SUMMARY:Ceia de Natal
DTSTART:20071224T223000Z
DURATION:PT3H
END:VEVENT
END:VCALENDAR
```

Figura 3.9: Exemplo de método PUBLISH para um evento

A figura 3.9 mostra um exemplo deste método. `PUBLISH` admite que componentes de mais de um tipo sejam incluídos em um mesmo objeto iCalendar. A figura 3.10 mostra

um exemplo deste método sendo utilizado para exportar todos os componentes de uma agenda em um único objeto iCalendar.

```
BEGIN:VCALENDAR
PRODID:-//Unicamp/iScheduler//EN
METHOD:PUBLISH
VERSION:2.0
...
BEGIN:VEVENT
UID:20071005T133225Z-00001@example.com
DTSTAMP:20071014T225108Z
ORGANIZER:mailto:A@example.com
SUMMARY:Ceia de Natal
DTSTART:20071224T223000Z
DURATION:PT3H
END:VEVENT
...
BEGIN:VTODO
UID:20080108T090000Z-00001@example.com
DTSTAMP:20080108T090000Z
SUMMARY:Declarar imposto de renda
DUE:20080430T110000Z
CATEGORIES:BUSINESS
PRIORITY:5
PERCENT-COMPLETE:50
END:VTODO
...
END:VCALENDAR
```

Figura 3.10: Exemplo de método PUBLISH para diversos componentes

REQUEST

O método `REQUEST` inicia um processo de negociação de agendamento através de um convite para os participantes de um evento ou to-do. Este método também é utilizado para reagendar um evento, mudando seus parâmetros, e para confirmar um evento já agendado.

A figura 3.11 mostra um exemplo deste método, que consiste em um convite de um indivíduo “A”, que é o organizador, aos participantes “B”, ‘C” e “D”. A inclusão da propriedade `ATTENDEE` correspondente ao indivíduo “E” na condição de não-participante,

```
BEGIN:VCALENDAR
PRODID:-//Unicamp/iScheduler//EN
METHOD:REQUEST
VERSION:2.0
BEGIN:VEVENT
UID:20071005T133225Z-00001@example.com
DTSTAMP:20071014T225108Z
ORGANIZER:mailto:A@example.com
ATTENDEE;PARTSTATUS=CONFIRMED:mailto:A@example.com
ATTENDEE;RSVP=TRUE:mailto:B@example.com
ATTENDEE;RSVP=TRUE:mailto:C@example.com
ATTENDEE;RSVP=TRUE:mailto:D@example.com
ATTENDEE;ROLE=NON-PARTICIPANT:mailto:E@example.com
SUMMARY:Reunião de departamento
DTSTART:20071224T223000Z
DURATION:PT3H
LOCATION:Sala 1
END:VEVENT
END:VCALENDAR
```

Figura 3.11: Exemplo de método REQUEST

expressa pelo parâmetro `ROLE`, corresponde à operação *Carbon copy* (Cc) em emails. Note que, através do valor do parâmetro `PARTSTATUS` para a propriedade `ATTENDEE` correspondente, o indivíduo "A" já confirma sua presença e, através do valor do parâmetro `RSVP`, uma resposta sobre a participação é requisitada aos demais participantes.

No contexto de componentes `VFREEBUSY`, o método `REQUEST` tem o papel de requisitar informações de *free-busy time* para um determinado período de tempo.

REPLY

O método `REPLY` é utilizado pelos participantes de um evento para aceitar ou rejeitar uma proposta de evento ou to-do.

A figura 3.12 mostra um exemplo deste método, no qual o participante "B" confirma sua presença no evento proposto pelo método `REQUEST` da figura 3.11, que é identificado pela propriedade `UID`. Caso o participante "B" desejasse rejeitar o convite para o evento, bastaria mudar o valor do parâmetro de status `PARTSTAT` para `DECLINED`.

No contexto de componentes `VFREEBUSY`, o método `REPLY` tem o papel de enviar informações de *free-busy time* sobre um determinado período de tempo. A RFC 2446 especifica que o tipo (`FBTYPE`) dos componentes `FREEBUSY` utilizados neste método não pode

```
BEGIN:VCALENDAR
PRODID:-//Unicamp/iScheduler//EN
METHOD:REPLY
VERSION:2.0
BEGIN:VEVENT
UID:20071005T133225Z-00001@example.com
DTSTAMP:20071011T145000Z
ORGANIZER:mailto:A@example.com
ATTENDEE;PARTSTAT=ACCEPTED;CN=B:mailto:B@example.com
COMMENT:Confirmado!
END:VEVENT
END:VCALENDAR
```

Figura 3.12: Exemplo de método REPLY

ser `FREE`, isto é, apenas informações sobre períodos de tempo não-livres são enviados e, a partir delas, os períodos de tempo livres devem ser inferidos.

ADD

O método `ADD` age incrementalmente, adicionando uma instância a um evento, to-do ou journal entry recorrente. Este método pode ser substituído por um método `PUBLISH` que republica as informações de todas as instâncias do componente de calendário em questão, incluindo aquela a ser adicionada. Embora mais verbosa, esta é uma alternativa equivalente, para a qual o método `ADD` oferece uma opção que pode poupar o tráfego desnecessário de grandes volumes de dados.

CANCEL

O método `CANCEL` cancela um evento já publicado, agendado, ou cujo agendamento esteja em negociação.[7]

A figura 3.13 mostra um exemplo deste método, com o qual o organizador "A" poderia cancelar o evento proposto pelo método `REQUEST` da figura 3.11. Seria possível cancelar o evento parcialmente, incluindo apenas as propriedades `ATTENDEE` correspondentes aos participantes para os quais o evento seria cancelado. Note a presença da propriedade `SEQUENCE` com valor 1, que é utilizada para indicar uma nova versão da descrição do evento.

[7]Este método também pode ser utilizado para cancelar apenas uma instância de um evento recorrente, identificada por seu horário de início através da propriedade `RECURRENCE-ID`.

```
BEGIN:VCALENDAR
PRODID:-//Unicamp/iScheduler//EN
METHOD:CANCEL
VERSION:2.0
BEGIN:VEVENT
UID:20071005T133225Z-00001@example.com
DTSTAMP:20071014T225108Z
SEQUENCE:1
ORGANIZER:mailto:A@example.com
ATTENDEE:mailto:A@example.com
ATTENDEE:mailto:B@example.com
ATTENDEE:mailto:C@example.com
ATTENDEE:mailto:D@example.com
ATTENDEE:mailto:E@example.com
END:VEVENT
END:VCALENDAR
```

Figura 3.13: Exemplo de método CANCEL

COUNTER e DECLINECOUNTER

O método `COUNTER` é utilizado para fazer uma contra-proposta durante uma negociação de agendamento, oferecendo uma descrição alternativa para o evento ou to-do em questão, e o método `DECLINECOUNTER` é utilizado para recusar uma contra-proposta de descrição alternativa.

A figura 3.14 mostra um exemplo destes métodos. Com `COUNTER`, o indivíduo "D" sugere ao organizador "A" uma opção alternativa para o evento proposto pelo método `REQUEST` da figura 3.11. Com `DECLINECOUNTER`, o organizador "A" rejeita a contra-proposta feita pelo indivíduo "D".

REFRESH

O método `REFRESH` é utilizado para requisitar a versão mais atual de um evento, to-do, ou journal entry.

A figura 3.15 mostra um exemplo deste método, através do qual o indivíduo "C" requisita ao organizador "A" a descrição mais atual do evento proposto pelo método `REQUEST` da figura 3.11.

```
BEGIN:VCALENDAR
PRODID:-//Unicamp/iScheduler//EN
METHOD:COUNTER
VERSION:2.0
BEGIN:VEVENT
UID:20071005T133225Z-00001@example.com
DTSTAMP:20071014T225108Z
ORGANIZER:mailto:A@example.com
ATTENDEE;RSVP=TRUE:mailto:A@example.com
ATTENDEE;RSVP=TRUE:mailto:B@example.com
ATTENDEE;RSVP=TRUE:mailto:C@example.com
ATTENDEE;PARTSTATUS=CONFIRMED:mailto:D@example.com
ATTENDEE;ROLE=NON-PARTICIPANT:mailto:E@example.com
SUMMARY:Reunião de departamento
DTSTART:20071224T223000Z
DURATION:PT3H
LOCATION:Sala 2
END:VEVENT
END:VCALENDAR
```

```
BEGIN:VCALENDAR
PRODID:-//Unicamp/iScheduler//EN
METHOD:DECLINECOUNTER
VERSION:2.0
BEGIN:VEVENT
UID:20071005T133225Z-00001@example.com
DTSTAMP:20071016T071200Z
ORGANIZER:mailto:A@example.com
END:VEVENT
END:VCALENDAR
```

Figura 3.14: Exemplo de métodos COUNTER e DECLINECOUNTER

```
BEGIN:VCALENDAR
PRODID:-//Unicamp/iScheduler//EN
METHOD:REFRESH
VERSION:2.0
BEGIN:VEVENT
UID:20071005T133225Z-00001@example.com
DTSTAMP:20071014T145000Z
ORGANIZER:mailto:A@example.com
ATTENDEE;CN=C:mailto:C@example.com
END:VEVENT
END:VCALENDAR
```

Figura 3.15: Exemplo de método REFRESH

Mecanismo de seqüenciamento

O protocolo iTIP conta com um mecanismo de seqüenciamento de mensagens que implementa um relógio lógico. Como, dependendo do protocolo de transporte utilizado, as mensagens podem chegar fora de ordem, sua ordenação é garantida através da utilização da propriedade `SEQUENCE` do objeto iCalendar, cujo valor padrão é 0.

A cada operação realizada pelos métodos `PUBLISH`, `REQUEST` ou `CANCEL`, que mudam as propriedades de um componente de calendário, o valor de seqüência deve ser incrementado. Em uma negociação de agendamento, este mecanismo de seqüenciamento torna obsoletas todas as mensagens com valor de seqüência menor ao da última mensagem recebida. Esta identificação das mensagens é feita através da propriedade `UID`, e utiliza o valor da propriedade `DTSTAMP` como critério de desempate para ordenar duas mensagens com mesmo valor de seqüência.

Por exemplo, em resposta ao método `REFRESH` ilustrado na figura 3.15, o organizador "A" poderia executar o método `REQUEST` da figura 3.16 para reagendar o evento que havia sido proposto pelo método `REQUEST` da figura 3.11; denotando a aceitação da contra-proposta feita pelo usuário "C" com o método `COUNTER` da figura 3.14. Note a utilização do valor de seqüência para indicar a nova versão do evento.

3.2.3 iMIP

O protocolo iMIP [21] é uma implementação *store-and-forward* do iTIP sobre email, consistindo em um conjunto de regras sobre como incluir objetos iCalendar em um email e como eles devem ser tratados pelas aplicações leitoras.

```
BEGIN:VCALENDAR
PRODID:-//Unicamp/iScheduler//EN
METHOD:REQUEST
VERSION:2.0
BEGIN:VEVENT
UID:20071005T133225Z-00001@example.com
DTSTAMP:20071017T225108Z
SEQUENCE:2
ORGANIZER:mailto:A@example.com
ATTENDEE;PARTSTATUS=CONFIRMED:mailto:A@example.com
ATTENDEE;RSVP=TRUE:mailto:B@example.com
ATTENDEE;RSVP=TRUE:mailto:C@example.com
ATTENDEE;PARTSTATUS=CONFIRMED:mailto:D@example.com
ATTENDEE:mailto:E@example.com
SUMMARY:Reunião de departamento
DTSTART:20071224T223000Z
DURATION:PT3H
LOCATION:Sala 2
END:VEVENT
END:VCALENDAR
```

Figura 3.16: Exemplo de método REQUEST em resposta a REFRESH

3.3 Os sistemas de segunda geração e a evolução dos protocolos

A segunda geração dos sistemas de agendamento é caracterizada por uma forte ênfase em usabilidade, e pela adoção, em especial, do padrão iCalendar. Alguns destes sistemas também são compatíveis com o formato vCalendar, ainda muito utilizado em dispositivos móveis.

Atualmente os usuários têm várias escolhas entre sistemas *desktop* populares, como Microsoft Outlook, Apple iCal (cuja interface é mostrada na figura 3.17), Lotus Notes, Oracle Calendar e Mozilla Sunbird.

Estes sistemas são desprovidos de quaisquer funcionalidades voltadas para o aumento da eficiência dos agendamentos, e são utilizados basicamente como agendas eletrônicas para o registro de eventos. Os usuários podem manter diversos calendários e neles registrar eventos e to-dos, recebendo alertas definidos com o uso de `VALARM`. Funcionalidades de importação e exportação de dados através de arquivos .ics, a menos de alguns problemas de compatibilidade devido a bugs e extensões proprietárias indevidas na implementação

Figura 3.17: Apple iCal

do padrão iCalendar, têm permitido que os usuários migrem de uma aplicação para outra sem muitos problemas.

Embora a especificação iMIP tenha tido algum sucesso inicial, inclusive com implementações interoperáveis, ela não é comumente usada. Apesar de viabilizar a troca de informações no formato iCalendar, oferecendo um mecanismo de transporte via emails, iMIP sofre das limitações que a arquitetura de emails apresenta para a execução de operações de compartilhamento de calendários e agendamento; por exemplo:

- iMIP não é um protocolo adequado para transmitir informações tão sensíveis às latências presentes na comunicação via email;
- se o usuário utiliza mais de um leitor de email, não existe um mecanismo para fazer o controle das requisições iTIP já processadas;
- o próprio leitor de emails deve implementar rotinas específicas para tratar as mensagens do protocolo, possivelmente encaminhando-as para uma aplicação de agendamento do usuário, o qual, caso contrário, receberá apenas um email com dados iCalendar incorporados, o que não faz muito sentido.

Estas e outras dificuldades foram enfrentadas, novamente, com o uso de mecanismos proprietários de comunicação, o que fez com que apenas em contextos corporativos as vantagens de funcionalidades de agendamento integradas fossem sentidas. O Microsoft Outlook, por exemplo, faz uso da biblioteca Messaging Application Programming Interface (MAPI) para se comunicar com servidores Microsoft Exchange.

Soluções Web, como Yahoo! Calendar e Google Calendar, capazes de mimetizar as funcionalidades de interação dos sistemas desktop graças à evolução das técnicas AJAX[8]

[8]AJAX (Asynchronous JavaScript and XML) é uma técnica de desenvolvimento para a criação de aplicações online, cuja transmissão de dados com o servidor ocorre de forma transparente para o usuário.

de desenvolvimento de aplicações para browsers, têm conquistado uma grande comunidade de usuários que, mesmo a partir de diferentes organizações, têm alternativas para compartilhamento de calendários e agendamento.

Nestes sistemas, as agendas dos usuários são armazenadas em servidores online, podendo ser compartilhadas e acessadas como um arquivo .ics disponibilizado via HTTP, ou visualizadas em páginas HTML. Extensões proprietárias para compartilhamento também foram adotadas. Por exemplo, usuários do Microsoft Outlook contam com uma aplicação Windows que permite sincronização com o Yahoo! Calendar, e conectores para acesso a agendas do Google Calendar estão sendo incorporados a aplicações como o Mozilla Sunbird e o Mozilla Thunderbird – um leitor de emails que suporta iMIP através de um *add-in* que oferece funcionalidades de calendários e agendamento integradas, o Mozilla Lighting. Tais formas de compartilhamento de calendários, entretanto, dependem da adoção de aplicações específicas, não funcionando em todos os ambientes.

Estas soluções também disponibilizam funcionalidades de negociação e agendamento de eventos, como envio de convites por email e controle das respostas dos participantes. Como estas facilidades são implementadas também através de mecanismos não-iTIP específicos, elas só podem ser utilizadas para agendamento em grupo se todos os usuários adotarem o mesmo sistema.

O fato do iMIP não ter o mesmo nível de adoção do padrão iCalendar e as limitações no uso de emails para compartilhamento de calendários e agendamento motivaram um grande esforço de desenvolvimento em direção a um protocolo em tempo real para iTIP.

3.3.1 Um protocolo de transporte em tempo real para iTIP

Antes de ter sido encerrado, o grupo CalSch trabalhou, por anos, na construção de um protocolo de tempo real totalmente novo para iTIP. Devido à sua complexidade, o desenvolvimento do Calendar Access Protocol (CAP)[9] foi abortado, mas, para registro histórico do trabalho até então desenvolvido, ele chegou a ser publicado como a RFC 4324 [65] na condição de protocolo experimental.

Egen [27] enumera diversas razões que tornam o estabelecimento de protocolos abertos para compartilhamento de calendários e agendamento tão difícil: ao contrário de outras aplicações, como email, não é possível desenvolver um protocolo que incorpore as melhores práticas das implementações existentes, pois há escolhas de projeto decisivas que levam a mecanismos de interoperabilidade bastante distintos. Há, ainda, uma série de problemas técnicos de resolução não-trivial que devem ser tratados pelo protocolo, como fusos-horários e horários de verão[10], a interpretação correta de regras para eventos recor-

[9] CAP tinha muitas similaridades com o protocolo de emails IMAP, também criticado por sua complexidade de implementação, que gerou problemas de interoperabilidade entre servidores.

[10] Por exemplo, não há um registro universal de fusos horários e de vigência de horários de verão, embora

rentes, e a busca por tempo livre em uma agenda.

Neste intervalo, outras alternativas foram buscadas para viabilizar o compartilhamento de informações de calendários. O protocolo Web Distributed Authoring and Versioning (WebDAV) [25], que estende as capacidades de publicação e recuperação de recursos do HTTP com funcionalidades de suporte a coleções de recursos e metadados, foi adotado pelo Apple iCal com grande sucesso. Esta adaptação consiste em mapear eventos iCalendar para recursos HTTP e modelar calendários e agendas como coleções WebDAV.

Tal abordagem mostrou-se mais direta do que a elaboração de um novo protocolo, e foi implementada também por outras aplicações de agendamento, como o Mozilla Calendar. Entretanto, ela não resolve todas as questões, pois tarefas típicas do domínio, como negociação de agendamentos e busca por tempo livre nas agendas de outras pessoas, têm características particulares não contempladas por WebDAV.

Com o fim do grupo CalSch, foi fundado, em 2004, o Calendaring and Scheduling Consortium (CalConnect), uma organização cujo objetivo é estimular o desenvolvimento e a adoção de padrões abertos para trocas de calendários e agendamento. Estão reunidos, no CalConnect, os maiores desenvolvedores de software e instituições de pesquisa, organizados em comitês técnicos que debatem os problemas dos protocolos já existentes e promovem novos padrões através do processo de desenvolvimento de especificações da IETF.

Com suporte do CalConnect, padrões para o uso do WebDAV no compartilhamento de calendários e agendamento pela Internet estão sendo debatidos, e já resultaram no desenvolvimento do protocolo Calendaring Extensions to WebDAV (CalDAV).

CalDAV é um protocolo cliente-servidor que implementa, separadamente, as funções de compartilhamento de calendários e agendamento. A funcionalidade `calendar-access` permite o compartilhamento de calendários através da extensão das coleções WebDAV, e já foi publicada como a RFC 4791 [20]. A especificação da extensão `calendar-schedule`, que suporta os processos de negociação e agendamento, está em curso, e seu draft [19] está sendo discutido. Basicamente, ela define duas coleções especiais, `inbox` e `outbox`, que guardam as mensagens iTIP correspondentes às propostas recebidas e enviadas durante a negociação de agendamento de eventos. Esta organização torna possível o acesso de um mesmo calendário por diversas aplicações, e possibilita casos de uso de delegação, nos quais um usuário administra a agenda de outro (e.g., uma secretária).

A emergência do protocolo CalDAV promete possibilitar o compartilhamento de calendários e a troca de mensagens de agendamento no formato iCalendar de forma padronizada. Diversos fabricantes de software (e.g., Apple, Google, Mozilla, Oracle e Yahoo) já se comprometeram publicamente a implementá-lo[11].

a base de dados `tz` (`http://www.twinsun.com/tz/tz-link.htm`) tenha se estabelecido como um padrão de fato ao ser adotada em diversos sistemas operacionais e plataformas.

[11] A Microsoft, historicamente mais avessa à implementação de padrões abertos de comunicação, é um dos mais novos membros do CalConnect, o que pode indicar que as próximas versões de suas aplicações já sejam compatíveis com CalDAV, embora até o momento não haja nenhum anúncio oficial neste sentido.

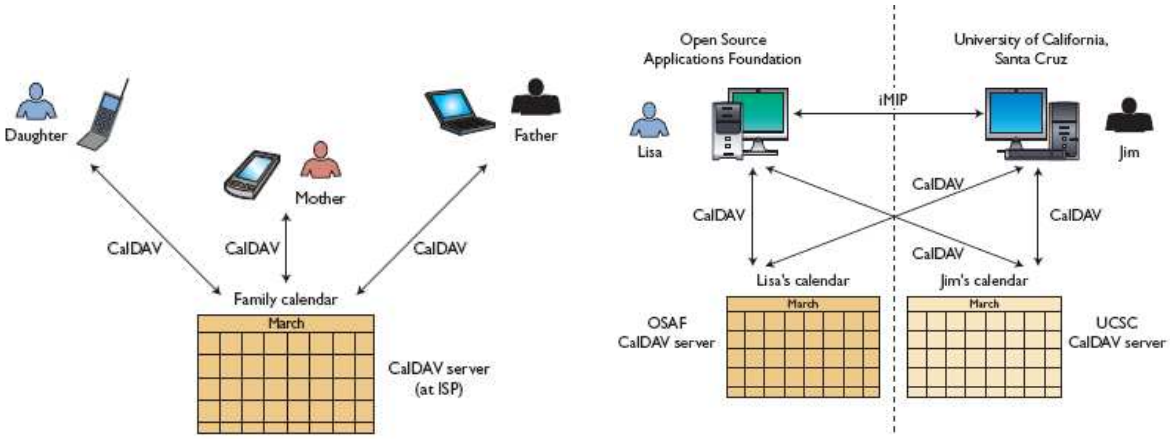


Figura 3.18: Integração entre dispositivos e aplicações distintas com CalDAV [26]

A figura 3.18 ilustra como o uso destas extensões poderá rapidamente propiciar interoperabilidade inclusive entre dispositivos heterogêneos de comunicação, que já fazem uso da pilha de protocolos HTTP. Diversos servidores CalDAV, inclusive de código aberto, estão sendo desenvolvidos[12]. Segundo o relatório mais recente dos testes de interoperabilidade do CalConnect [28], já há um nível razoável de interoperabilidade entre servidores e clientes com as funcionalidades básicas de compartilhamento de calendários, enquanto que a implementação das extensões de agendamento ainda é incipiente na maioria dos projetos, mesmo porque elas ainda estão sendo desenvolvidas.

3.3.2 Evolução dos protocolos já existentes

O consórcio CalConnect também organiza testes de interoperabilidade para verificar a aderência aos padrões e a existência de problemas de interoperabilidade entre implementações dos protocolos, não somente para CalDAV mas também para as especificações já existentes. Em parceria com o grupo de trabalho Calendaring and Scheduling Standards Simplification (Calsify) da IETF, estão sendo feitas a revisão e simplificação dos padrões iCalendar, iTIP e iMIP.

Os drafts para as versões "bis" destes protocolos [23, 17, 51] estão em discussão e em breve deverão ser publicados como novas RFCs, que deverão resolver os problemas de compatibilidade e pontos de simplificação identificados pela experiência de mais de uma década na implementação destes protocolos[13]. A versão "bis" da RFC 2445 determina, ainda, regras mais claras para a extensão do padrão iCalendar; obrigando, por exemplo, que adições, exclusões e modificações de componentes sejam realizadas através da

[12] `http://ietf.osafoundation.org/caldav/homepage/projects.html`.

[13] A lista com os *issues* encontrados nas RFCs atuais pode ser acessada no endereço `http://www.ofcourseimright.com/cgi-bin/roundup/calsify/`.

publicação de novas RFCs.

Devido à ênfase do grupo de trabalho Calsify em simplificar e fazer com os padrões já existentes funcionem, iniciativas como o desenvolvimento de um esquema de representação de dados iCalendar com XML (xCalendar[14] e xCal[15]) foram deixadas fora de seu escopo, pois implicariam em *mudanças* no formato de dados padrão.

3.4 Limitações dos protocolos

Os padrões iCalendar e iTIP têm uma séria limitação: eles não permitem o uso de mais de uma proposta alternativa durante a negociação de agendamento de um evento – i.e., a aplicação da estratégia múltipla de anúncio e da estratégia de resposta com alternativas, descritas na seção 2.5.1.

Não há uma forma de propor um conjunto de horários de início para um evento. Esta inflexibilidade pode também representar um alto custo de comunicação. No pior caso, se somente a n-ésima proposta de evento enviada pelo organizador for aceitável para todos os participantes, oferecer uma descrição de evento por proposta implica em n rounds de negociação (e n conjuntos de mensagens trocados). Da mesma forma, no melhor caso, se o organizador tivesse proposto n descrições alternativas de um evento, somente um round de negociação teria sido suficiente. Isto se considerarmos que, ao final dos n rounds, os *slots* de tempo inicialmente disponíveis não tenham sido ocupados por outras negociações de agendamento concorrentes.

Sayers e Letsinger [68] discutem, de uma forma mais abrangente, outras deficiências da ontologia representada pelo padrão iCalendar: ela usa uma estrutura "linear" para descrever eventos, o que não favorece a representação de informações incompletas. Embora isto não seja um grande problema para o compartilhamento de calendários, quando todos os dados a respeito de um evento são previamente conhecidos, esta limitação tem um peso fundamental durante negociações de agendamento.

Uma negociação de agendamento, tipicamente, é iniciada com uma requisição similar a "Gostaria de encontrar-me com você em Campinas na terça-feira pela manhã, ou em São Paulo na sexta-feira à tarde", para a qual uma resposta razoável seria "Estou disponível a qualquer hora na terça-feira, mas preferiria um horário entre 10h e 11h". Entretanto, o formato iCalendar não oferece formas para expressar as incertezas e preferências que fazem parte do fluxo normal de informações que precede o agendamento de um evento.

Na ontologia CoolAgent, Sayers e Letsinger [67] propõem que um evento seja visto como uma combinação de instâncias compostas de três dimensões: pessoas, lugares e tempo. Estas entidades podem ser organizadas de forma flexível em uma estrutura hierárquica e

[14] `http://tools.ietf.org/html/draft-hare-xcalendar-03`.
[15] `http://tools.ietf.org/html/draft-royer-calsch-xcal-03`.

recursiva que permite relacionamentos conjuntivos (`all-of`) e disjuntivos (`one-of`), conforme mostrado na figura 3.19.

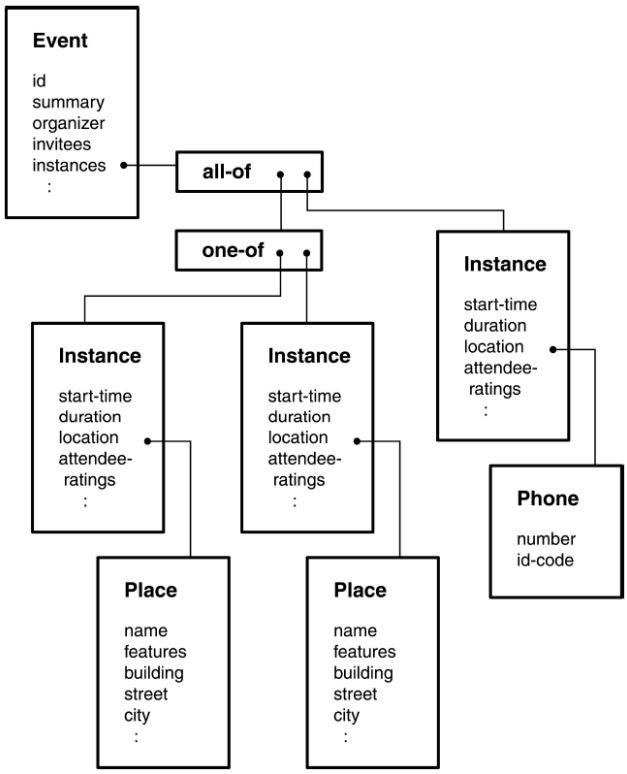


Figura 3.19: A flexível estrutura hierárquico-recursiva da ontologia CoolAgent [67]

Embora tenha uma forma pouco expressiva para representar as entidades lugar (propriedade `LOCATION`) e tempo (propriedades `DTSTART` e `DTEND` – ou `DURATION`), o formato iCalendar tem alguma flexibilidade na representação dos participantes de um evento. É possível, por exemplo, adicionar um atributo `ROLE` para cada parâmetro `ATTENDEE` informando o caráter da participação da pessoa no evento, como mostra o exemplo da figura 3.20, mas relações mais complexas, do tipo `one-of`, que se aplicam sobre um grupo de participantes, não podem ser representadas.

```
ATTENDEE;ROLE=CHAIR:mailto:A@example.com
ATTENDEE;ROLE=REQ-PARTICIPANT:mailto:B@example.com
ATTENDEE:ROLE=OPT-PARTICIPANT:mailto:C@example.com
ATTENDEE:ROLE=NON-PARTICIPANT:mailto:D@example.com
ATTENDEE;ROLE=REQ-PARTICIPANT:mailto:Room@example.com
```

Figura 3.20: Exemplo de papéis dos participantes de um evento iCalendar

O problema de representar lugares como entidades, e não como parâmetros de eventos, está em vias de ser resolvido com a proposta do componente `VVENUE` [59], específico para esta finalidade.

Apesar de os X-tokens definidos pela RFC 2445 serem um recurso adicional para a representação de dados não previstos no padrão iCalendar, eles só podem ser utilizados no escopo de uma mesma aplicação, não tendo utilidade para a implementação de uma semântica mais poderosa como a proposta na ontologia CoolAgent.

3.5 Agendamento em grupo como um problema de otimização

Paralelamente à popularização das aplicações pessoais para compartilhamento de calendários e agendamento, a idéia de aplicar técnicas de otimização para construir agendamentos mais eficientes tornou-se comum. Este conceito originou um grupo de sistemas orientados à alocação ótima de recursos classificados por Pinedo e Chao [63] em: sistemas comerciais, sistemas de aplicação específica e protótipos acadêmicos.

Sistemas comerciais não são projetados para um usuário específico, mas concebidos para se ajustarem a um número grande de usuários. Geralmente eles concentram-se em uma abordagem particular de agendamento, encarando-o como um problema de alocação de recursos e atribuição de tarefas. As principais entidades destas aplicações não são eventos, mas projetos compostos de fases e tarefas a serem executados dentro de um prazo determinado. Estas tarefas podem ter relações de precedência entre si, e têm uma demanda de recursos que deve ser administrada. O Microsoft Project é o sistema mais popular nesta categoria.

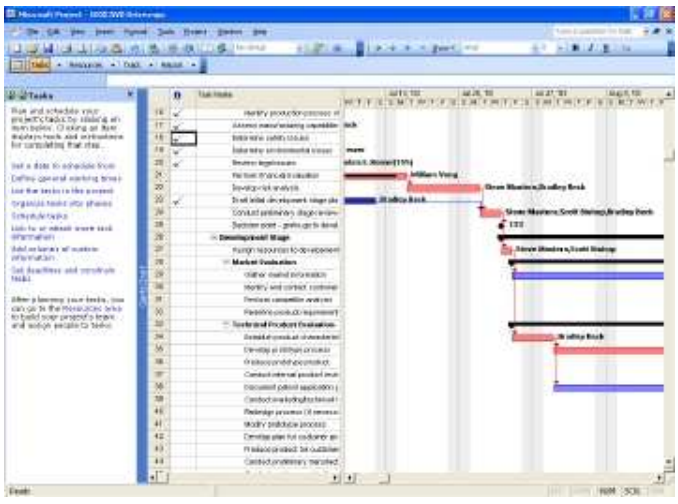


Figura 3.21: O uso de gráficos de Gantt na interface do Microsoft Project

Embora o tempo das pessoas possa ser mapeado como recursos, os sistemas de agendamento baseados em gerência de projetos são apropriados para planejamento de médio e longo prazo. Eles não são adequados para agendamento em grupo em escala diária (e.g., para negociar o agendamento de uma reunião). Embora gerentes de projetos e planejadores sejam capazes de aproveitar estas ferramentas de forma bastante útil, elas dificilmente poderiam ser adaptadas para utilização pelo "usuário médio" de aplicações desktop, que precisaria apreender conceitos de gerência de projetos e adaptar suas práticas de trabalho. Por exemplo, no Microsoft Project e em seus equivalentes, o uso do tempo é tipicamente acompanhado com o uso de gráficos de Gantt, como mostra a figura 3.21, em contraste com o uso da metáfora de calendários presente nos sistemas tradicionais de agendamento.

Sistemas de aplicação específica são ainda mais elaborados, e projetados para atividades de agendamento bastante específicas. Esse tipo de sistema requer um alto grau de capacitação e especialização dos usuários, sendo utilizado predominantemente em empresas de grande porte capazes de sustentar seus altos custos de desenvolvimento, manutenção e uso. Esses sistemas são utilizados para a resolução de problemas cujas restrições e o critério de otimalidade são bem definidos, como roteamento de veículos e planejamento de linhas de produção em manufatura.

Protótipos acadêmicos (e.g., [47]), por sua vez, desenvolvem e exercitam determinadas técnicas algorítmicas que têm um potencial – ainda não plenamente explorado – de influenciar a concepção dos outros sistemas de maior alcance. A série de conferências "Practice and Theory of Automated Timetabling" (PATAT)[16], por exemplo, tem dado uma inquestionável contribuição para a compreensão e o tratamento de problemas reais de grande porte, como geração de escalas de trabalho (*rostering*), construção de grades de horários para escolas e universidades (*timetabling*), e agendamento de ônibus e trens [85]. Contudo, a pesquisa no campo da otimização tem, em sua maior parte, privilegiado estratégias de modelagem e resolução para gerar soluções de agendamento que melhor satisfaçam as complexas restrições que tornam estes problemas de difícil resolução.

Entretanto, o problema de agendamento em grupo não é um problema bem definido como estes. As únicas restrições "fortes" são: (1) eventos simultâneos não podem ser agendados para a mesma pessoa, e (2) as datas máximas de agendamento dos eventos devem ser respeitadas. Mas há incontáveis restrições "fracas", implícitas – como preferências de tempo – que os usuários desejam satisfazer durante o agendamento. Independentemente de ser possível, especificar tais condições explicitamente em termos de restrições de agendamento seria extremamente oneroso, indo contra a necessidade de ferramentas simples e eficazes. Adicionalmente, para agendamento em grupo, o critério de otimalidade também não é claro. Como os eventos são agendados na medida em que são propostos e têm seus horários de início negociados, critérios de otimização como aqueles usados em sistemas de aplicação específica (e.g., makespan) não fazem sentido.

[16] `http://www.asap.cs.nott.ac.uk/patat/patat-index.shtml`

3.5.1 Timetable Rearrangement

Sugihara et al. [77] notam que o real problema de otimização para agendamento em grupo não consiste em gerar agendamentos ótimos, mas em, diante de um novo evento a ser agendado, *rearranjar* o agendamento já existente de forma a minimizar as mudanças neste. Não é possível realocar todos os eventos a cada novo agendamento para satisfazer um critério de otimalidade, pois, na prática, remarcar um evento já agendado é uma tarefa inconveniente que traz implicações sociais e deve ser evitada.

Define-se como Timetable Rearrangement (TR) o problema de otimização cujo objetivo é minimizar o número de mudanças necessárias para agendar um novo evento caso não haja horários livres para ele.

Definição do problema [77]

Seja n um inteiro positivo. Um agendamento de n eventos é uma 8-tupla $T(n) = (P, M_n, <, t, p, w, \tau, \rho)$, onde

1. $P = \{1, 2, ..., m\}$ é um conjunto de m pessoas,
2. $M_n = \{1, 2, ..., n\}$ é um conjunto de n eventos,
3. $<$ é uma ordem parcial em M_n,
4. $t(m_i)$ é o tempo de duração do evento m_i,
5. $p(m_i)$ é um conjunto de grupos de pessoas $\{g_1, ..., g_r\}$ tal que exatamente uma pessoa em cada grupo deve participar do evento m_i,
6. $w(m_i)$ é um conjunto de instâncias de tempo nas quais o evento m_i pode começar,
7. $\tau(m_i)$ é o horário de início do evento m_i, e
8. $\rho(m_i)$ é o conjunto de participantes de m_i,

que obedece às seguintes condições:

1. nenhuma pessoa participa de mais do que um evento simultaneamente;
2. se $m_i < m_j$, então m_j começa após o término de m_i;
3. para qualquer evento m_i e qualquer grupo $g \in p(m_i)$, exatamente uma pessoa em g participa do evento m_i; e
4. cada evento m_i começa em uma instância de tempo em $w(m_i)$.

Em um agendamento $T(n)$, $<$, t, p e w representam os requisitos dos eventos. Os parâmetros τ e ρ representam um agendamento de eventos que satisfaz todas as condições 1-4. O agendamento de outros recursos, como salas de reuniões, também pode ser incorporado com a utilização de pseudo-pessoas. Por exemplo, a seleção de uma sala para um evento m_i dentre um conjunto de salas S_1, S_2, ..., S_k é representada como um grupo de pseudo-pessoas em $p(m_i)$.

Exemplo 1 Consideremos o seguinte agendamento $T(5)$ de 5 eventos, mostrado na figura 3.22.

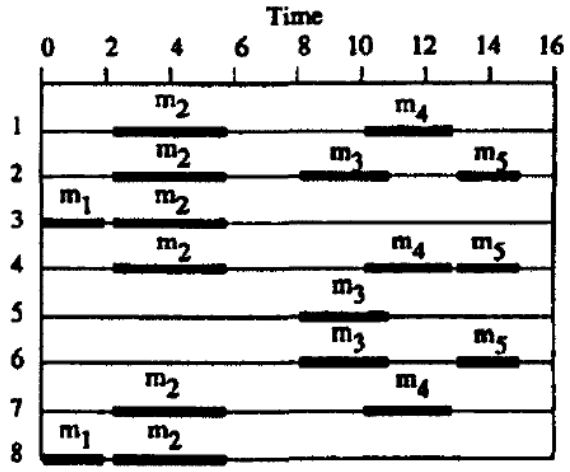


Figura 3.22: Representação gráfica do agendamento $T(5)$ [77]

$P = \{1, 2, 3, 4, 5, 6, 7, 8\}$
$M_5 = \{m_1, m_2, m_3, m_4, m_5\}$

$m_1 < m_5$ e $m_2 < m_4$
$t(m_1) = 2$, $t(m_2) = 4$, $t(m_3) = 3$, $t(m_4) = 3$, $t(m_5) = 2$

$p(m_1) = \{\{2, 3\}, \{7, 8\}\}$
$p(m_2) = \{\{1\}, \{2\}, \{3\}, \{4\}, \{7\}, \{8\}\}$
$p(m_3) = \{\{2, 3\}, \{7, 8\}\}$
$p(m_4) = \{\{2, 3\}, \{7, 8\}\}$
$p(m_5) = \{\{2, 3\}, \{7, 8\}\}$

$w(m_1) = \{0, 1, 2, 3, 4\}$
$w(m_2) = \{2, 3\}$
$w(m_3) = \{2, 3, 8, 9\}$

$w(m_4) = \{k \mid 2 \leq k \leq 12\}$
$w(m_5) = \{k \mid 0 \leq k \leq 16\}$

$\tau(m_1) = 0$, $\tau(m_1) = 2$, $\tau(m_1) = 8$, $\tau(m_1) = 10$ e $\tau(m_1) = 13$

$\rho(m_1) = \{3, 8\}$
$\rho(m_2) = \{1, 2, 3, 4, 7, 8\}$
$\rho(m_3) = \{2, 5, 6\}$
$\rho(m_4) = \{1, 4, 7\}$
$\rho(m_5) = \{2, 4, 6\}$

O problema TR é definido da seguinte forma: dados como parâmetros de entrada

- um agendamento $T(n) = (P, M_n, <, t, p, w, \tau, \rho)$,
- um novo evento m_{n+1} a ser agendado,
- $t(m_{n+1})$, $p(m_{n+1})$ e $w(m_{n+1})$,
- uma ordem parcial $<'$ em $M_{n+1} = M_n \cup \{m_{n+1}\}$, e
- um conjunto $F(\subseteq M_n)$ de eventos cujos horários de início estão fixados,

encontrar um agendamento $T(n+1) = (P, M_{n+1}, <', t, p, w, \tau', \rho')$ que minimiza a seguinte função objetivo:

$$z = \sum_{m_i \in M_n \ e \ \tau(m_i) \neq \tau'(m_i)} |\rho(m_i) \cup \rho'(m_i)| + \sum_{m_i \in M_n \ e \ \tau(m_i) = \tau'(m_i)} |(\rho(m_i) \cup \rho'(m_i)) - (\rho(m_i) \cap \rho'(m_i))| \quad (3.1)$$

sujeita à restrição $\tau(m_i) = \tau'(m_i) \ \forall \ m_i \in F$.

A função objetivo z representa o número total de pessoas que devem mudar suas agendas devido ao rearranjo dos eventos. O primeiro termo de z corresponde ao número de pessoas participantes de eventos cujos horários de início são modificados. O segundo termo corresponde ao número de pessoas que deixam de participar de um evento ou passam a participar de um evento cujo horário de início não é modificado. A restrição obriga que o rearranjo não afete o agendamento dos eventos em F.

Exemplo 2 Consideremos a seguinte instância de TR:

- o agendamento $T(5)$ do exemplo 1;
- um novo evento m_6;

- $t(m_6) = 3$, $p(m_6) = \{\{1\}, \{3, 4\}, \{6\}\}$ e $w(m_6) = \{8, 9, 10, 11, 12\}$;
- $m_1 <' m_5$, $m_2 <' m_4$ e $m_3 <' m_6$;
- $F = \{m_5\}$.

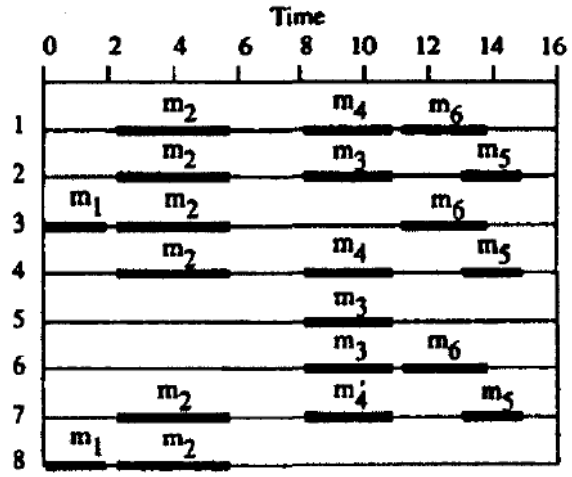


Figura 3.23: Representação gráfica do agendamento $T(6)$ [77]

A figura 3.23 mostra um agendamento ótimo $T(6) = (P, M_6, <', t, p, w, \tau', \rho')$. Nesta solução, $\tau'(m_4) = 8$, $\rho'(m_5) = \{2, 4, 5\}$, $\tau'(m_6) = 11$ e $\rho'(m_6) = \{1, 3, 6\}$. Note que o horário de início de m_4 é modificado de 10 para 8, e o participante 6 é substituído pelo participante 7 em m_5. Portanto, o valor da função objetivo é[17]:

$$\begin{aligned} z &= |\rho(m_4) \cup \rho'(m_4)| \; + \; |(\rho(m_5) \cup \rho'(m_5)) - (\rho(m_5) \cap \rho'(m_5))| \\ &= 3 + 2 \\ &= 5. \end{aligned}$$

Sugihara et al. [77] demonstram que TR é NP-difícil, e que o problema de factibilidade de TR, isto é, determinar se existe uma solução factível para uma dada instância de TR, é NP-completo[18]. Isto significa que não existe um algoritmo de aproximação de tempo polinomial para TR a menos que P = NP.

[17]Na verdade, esta definição da função objetivo tem uma imprecisão, pois conta duas vezes a mesma pessoa se ela participar de um evento cujo horário foi modificado *e* deixar de participar ou passar a participar de um outro evento, o que neste caso ocorre com a pessoa 7.

[18]Detalhadas demonstrações a respeito da complexidade de problemas de *timetabling* podem ser encontradas em [12].

Busca exaustiva

Sugihara et al. [77] apresentam um algoritmo de busca exaustiva para a resolução de TR. Este algoritmo é baseado em uma versão simplificada de TR – Restricted Timetable Rearrangement (RTR). Em uma instância RTR, as seguintes restrições adicionais são impostas ao rearranjo de TR:

1. nenhum participante de cada evento em M_n é modificado; e
2. uma ordem implícita de M_n em $T(n)$ é preservada. Isto é, se uma pessoa participa de dois eventos m_i e m_j tal que $\tau(m_i) < \tau(m_j)$, então $\tau'(m_i) < \tau'(m_j)$.

Denota-se por RTR(x,X) uma instância de RTR na qual há apenas uma escolha possível x para o horário de início e X para o conjunto de participantes do evento m_{n+1}. Como a restrição 1 implica que $\rho'(m_i) = \rho(m_i)$ para todo evento m_i, a função objetivo z em 3.1 é simplificada para

$$z = \sum_{m_i \in M_n \text{ e } \tau(m_i) \neq \tau'(m_i)} |\rho(m_i)|. \tag{3.2}$$

Seja $X = \{\{p_1\}, \{p_2\}, ..., \{p_k\}\}$ e $B_i(x)$ um conjunto ordenado de eventos $\{m_{i,1}$, $m_{i,2}$, ..., $m_{i,j}$, $m_{i,j+1}$, ..., $m_{i,b}\}$ dos quais a pessoa p_i participa durante um intervalo de tempo $(x, x + t(m_{n+1}))$ no agendamento corrente $T(n)$. A ordem em $B_i(x)$ é definida pela ordem ascendente dos horários de início dos eventos. Seja $<_n$ a ordem parcial em M_n induzida por um agendamento $T(n) = (P,\ M_n,\ <,\ t,\ p,\ w,\ \tau,\ \rho)$, definida da seguinte forma: $m_i <_n m_j \Leftrightarrow m_i < m_j \vee (\rho(m_i) \cap \rho(m_j) \neq \emptyset \wedge \tau(m_i) < \tau(m_j))$.

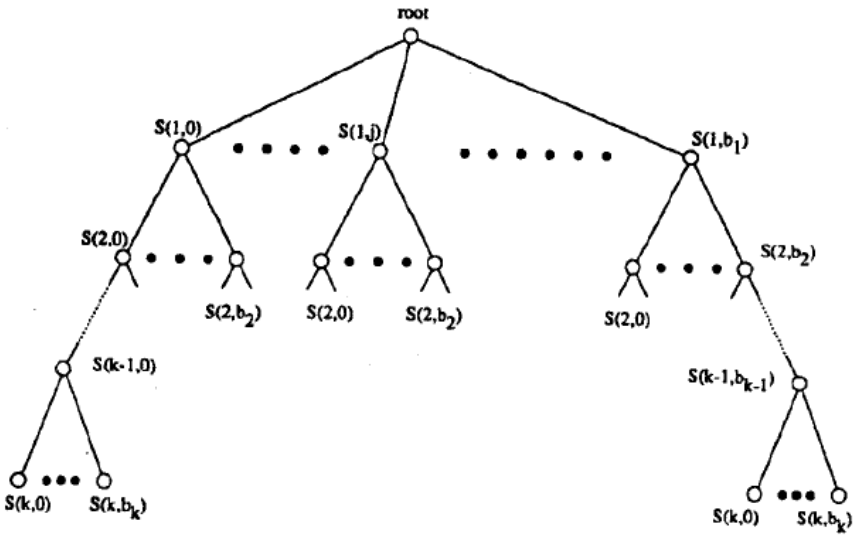


Figura 3.24: Uma árvore de busca para RTR(x,X) [77]

O algoritmo para RTR(x,X) consiste em uma busca exaustiva, cuja árvore é mostrada na figura 3.24. Cada nó é identificado pela operação executada quando o nó é visitado. A operação $S(i, j)$ é definida como uma seqüência de dois procedimentos $LSHIFT(m_{i,j}, x)$ e $RSHIFT(m_{i,j+1}, x)$. Se $B_i(x) \neq \emptyset$, $LSHIFT(m_{i,j}, x)$ move, em ordem, cada s-ésimo evento ($s \leq j$) em $T(n)$ "à esquerda", e $RSHIFT(m_{i,j+1}, x)$ move, em ordem, todos os outros eventos em $B_i(x)$ em $T(n)$ "à direita", de modo que a pessoa p_i não participe de nenhum evento durante o intervalo $(x, x + t(m_{n+1}))$.

Uma solução factível é obtida em uma folha da árvore de busca. O agendamento ótimo é aquele com o menor valor de z (3.2) entre as soluções factíveis. Este algoritmo tem complexidade de tempo $O(m^2n^2c^k)$, onde m é o número de pessoas, n é o número de eventos, $c = \max\{|B_i(x)| \mid p_i \in X\} + 1$ e $k = |X|$.

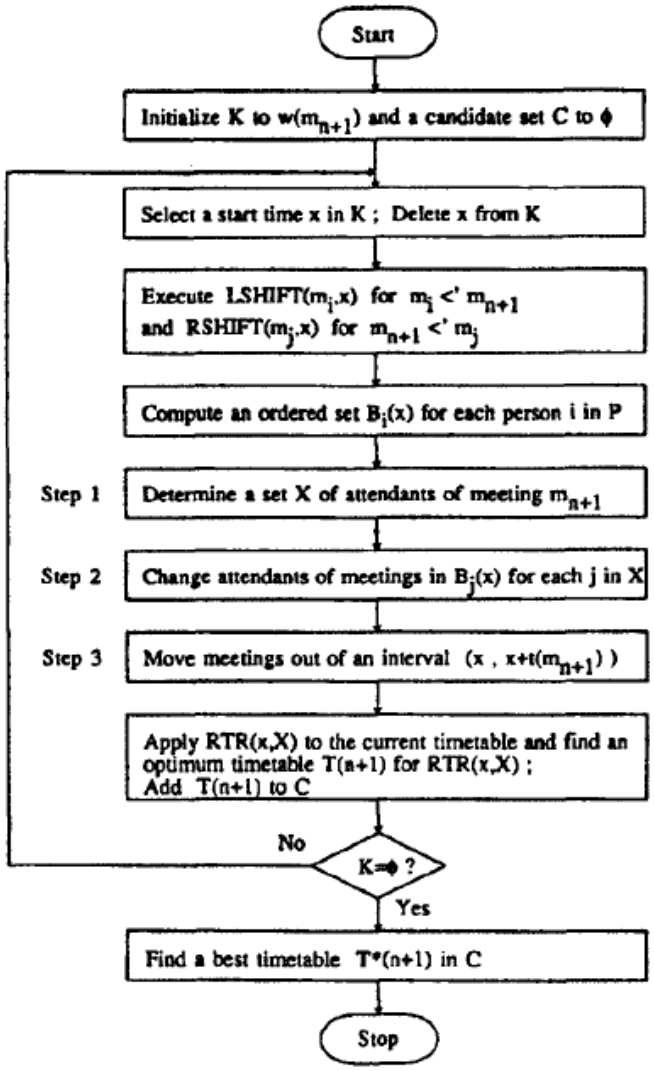


Figura 3.25: Algoritmo com heurística para TR [77]

O algoritmo heurístico para TR baseado no algoritmo para RTR(x,X) é mostrado na figura 3.25. Os passos 1, 2 e 3 constituem uma rotina de pré-processamento do algoritmo RTR para cada horário de início possível $x \in w(m_{n+1})$. No passo 1, um conjunto X de participantes de m_{n+1} é selecionado e fixado. No passo 2, se um participante $j \in X$ de um evento $m_i \in M$ pode ser substituído por um outro participante $k \notin X$ sem mudar o agendamento de nenhum outro evento em M_n, então j é substituído por k. No passo 3, se um evento $m_i \in B_i(x)$ para algum participante $j \in X$ pode ser movido fora do intervalo de tempo $(x, x+t(m_{n+1}))$ sem mudar o agendamento dos eventos em M_n, então m_i é movido. Estes passos tem o objetivo de melhorar as soluções obtidas e diminuir a complexidade de resolver RTR(x,X).

A complexidade desta busca exaustiva é $O(m^2n^2c^k\omega^2)$, onde $\omega = \max\{|w(m_i)| \mid 1 \leq i \leq n+1\}$, sendo exponencial no número k de participantes do evento m_{n+1} no pior caso.

Busca A*

Sugumaran et al. [78] desenvolveram um algoritmo mais eficiente para o problema TR, que utiliza uma estratégia de busca A*.

Seja $p(m_{n+1}) = g_1, ..., g_r$ o conjunto de grupos de pessoas a serem consideradas para o agendamento do novo evento m_{n+1}. Seja $\rho(m_{n+1})$ o conjunto de participantes para m_{n+1} escolhidos de $p(m_{n+1})$. E seja $(x, x + t(m_{n+1}))$ o intervalo de tempo considerado para o agendamento de m_{n+1}. São definidas as seguintes operações utilizadas na busca A* para TR, ilustradas na figura 3.26.

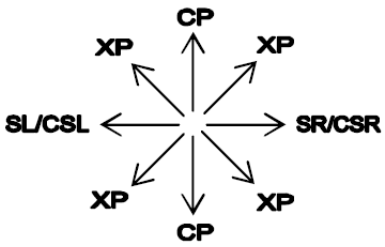


Figura 3.26: Operações utilizadas na busca A* [79]

- *Shift Left* (SL): move eventos horizontalmente à esquerda. Para cada pessoa $j \in \rho(m_{n+1})$ e cada evento $m_i \in (x, x + t(m_{n+1}))$, se há um intervalo de tempo $(s, s + t(m_{n+1}))$ à esquerda de x e disjunto de $(x, x+t(m_{n+1}))$ tal que $s \in w(m_i)$ e nenhuma pessoa em $\rho(m_i)$ participa de qualquer evento durante este intervalo, o horário de início de m_i é mudado para s.

- *Shift Right* (SR): move eventos horizontalmente à direita. Para cada pessoa $j \in \rho(m_{n+1})$ e cada evento $m_i \in (x, x + t(m_{n+1}))$, se há um intervalo de tempo $(s, s +$

$t(m_{n+1}))$ à direita de x e disjunto de $(x, x + t(m_{n+1}))$ tal que $s \in w(m_i)$ e nenhuma pessoa em $\rho(m_i)$ participa de qualquer evento durante este intervalo, o horário de início de m_i é mudado para s.

- *Change Person* (CP): troca uma pessoa verticalmente dentro de um grupo, liberando slots de tempo para as pessoas $\rho(m_i)$ em $(x, x + t(m_{n+1}))$ dentro de um grupo. Para cada evento $m_i \in (x, x + t(m_{n+1}))$, se nenhuma pessoa em $\rho(m_i)$ com exceção de uma pessoa $j \in \rho(m_i)$ está associada ao evento m_i, e há uma pessoa k em $\{a \mid a, j \in g_r \text{ e } g_r \in p(m_i), k \neq j\}$ que não participa de nenhum evento durante $(\tau(m_i), \tau(m_i) + t(m_i))$, o participante de m_i é mudado de j para k.

- *eXchange Persons* (XP): troca pessoas dentro de um grupo em diferentes eventos, liberando slots de tempo para o evento a ser agendado através de mudanças diagonais. Para quaisquer duas pessoas $j, k \in g_p$ em $p(m_i)$ e $j, k \in g_q$ em $p(m_n)$ tal que $j \in \rho(m_i)$ e $k \in \rho(m_n)$, j e k são trocados entre $\rho(m_i)$ e $\rho(m_n)$ se j está livre em $(x, x + t(m_n))$ e k está livre em $(x, x + t(m_i))$.

- *Continuous Shift Left* (CSL): move os eventos em $(x, x + t(m_{n+1}))$, bem como os eventos à esquerda deste intervalo, mais para a esquerda, de modo que cada pessoa $j \in \rho(m_{n+1})$ fique livre em $(x, x + t(m_{n+1}))$.

- *Continuous Shift Right* (CSR): move os eventos em $(x, x + t(m_{n+1}))$, bem como os eventos à direita deste intervalo, mais para a direita, de modo que cada pessoa $j \in \rho(m_{n+1})$ fique livre em $(x, x + t(m_{n+1}))$.

A função de avaliação $f(n) = g(n) + h(n)$ é computada com $g(n)$ sendo o custo de caminho desde a raiz até o nó, isto é, o número de participantes cujas agendas já foram alteradas, e a função heurística $h(n)$ consistindo no menor número de slots indisponíveis para o intervalos de tempo $(x, x + t(m_{n+1}))$ com $x \in w(m_{n+1})$.

A figura 3.27 mostra a árvore de busca para a instância TR do exemplo 2 da seção 3.5.1, iniciada no nó que representa $T(5)$. O valor à esquerda do nó indica a ordem em que ele é visitado na busca, e os valores à direita correspondem, respectivamente, a $h(n)$ e $g(n)$. A busca termina com a solução ótima $T(6)$ já apresentada, na qual o horário de início de m_4 é modificado de 10 para 8, e o participante 6 é substituído pelo participante 7 em m_5. Nós que não chegam a ser expandidos também são mostrados para efeito de comparação com algoritmos exaustivos, que computam todas as soluções e escolhem a melhor obtida.

Sugumaran et al. [80] mostram que a função heurística $h(n)$ é admissível, e que a aplicação sucessiva das operações SR, SL, CP, XP, CSL e CSR na busca A* leva ao nó que contém a solução ótima.

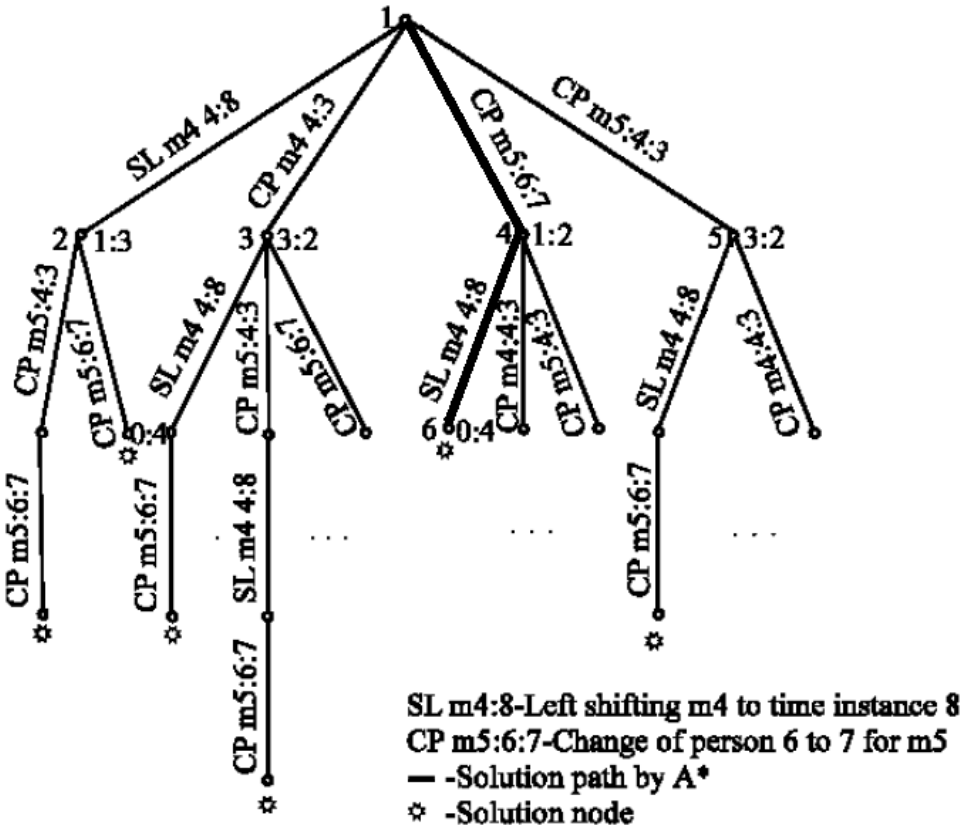


Figura 3.27: Árvore de busca A* (Adaptado de [80])

3.6 Agendamento em grupo como um problema de inteligência artificial

Da mesma forma que a otimização, técnicas de inteligência artificial (IA) também têm sido propostas para a resolução de problemas de agendamento [86].

Diversos métodos foram utilizados, por exemplo, para sugerir horários de agendamento de novos eventos com base em agendamentos passados, o que pode ser útil quando várias opções estão disponíveis. Neste caso, os benefícios do emprego de técnicas de aprendizado são perceptíveis [1]. Kozierok e Maes [45] propuseram há muito um agente de interface para agendamento que iria sugerir horários para a marcação de novos eventos.

O Calendar Apprentice descrito por Mitchell et al. [52] tem um escopo mais amplo: ele tenta predizer outras características de um evento, como local e participantes. Blum [8] demonstrou que, se as características a serem preditas forem criteriosamente selecionadas, o nível de acerto do algoritmo pode ser alto o suficiente para sua aplicação em um ambiente real.

Mueller [56] propõe o sistema SensiCal, mostrado na figura 3.28: um calendário com

"senso comum" que teria uma base de conhecimentos gerais sobre o mundo e seria apto a, por exemplo, advertir um usuário que convidasse uma pessoa vegetariana para uma churrascaria.

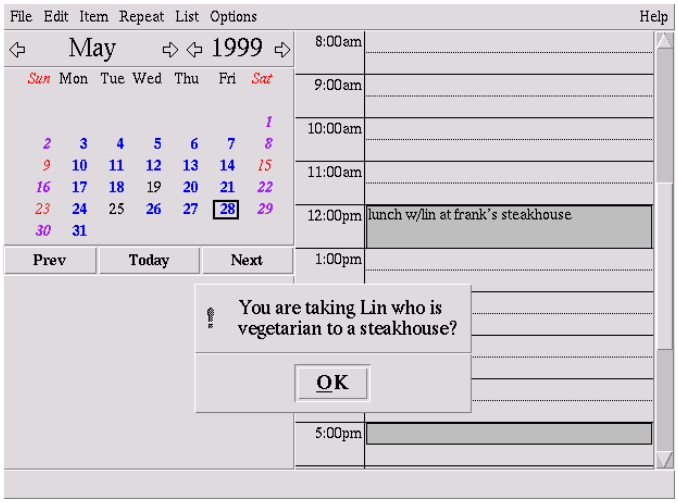


Figura 3.28: A base de dados de senso comum do SensiCal em ação

Para minimizar os problemas trazidos pela necessidade de reagendamentos, Modi e Veloso [53] sugerem um mecanismo de estimação da dificuldade de agendar um compromisso com outras pessoas, de forma a determinar qual rearranjo de eventos seria o mais adequado. Shintani et al. [72] propõem um mecanismo de revisão de preferências que adapta as preferências locais do usuário de forma que suas propostas de agendamento sejam mais facilmente aceitas por outros.

Sen [70] detalha a arquitetura de um sistema de agentes inteligentes que conduziriam autonomamente a negociação de agendamentos, satisfazendo as preferências dos usuários. Tal sistema pressupõe a capacidade de aprender essas preferências. Oh e Smith [60], por exemplo, usam classificadores para decidir se um evento deve ser cancelado ou não.

Novamente, no campo da inteligência artificial, a construção de mecanismos de agendamento utilizáveis em um ambiente de produção não tem sido enfatizada, e sim a aplicação de técnicas de aprendizado e inferência baseadas na disponibilidade de grandes volumes de informações. Embora a maioria destas propostas seja extremamente interessante, algumas são muito ambiciosas para aplicação em problemas reais, pois requerem dados que provavelmente não serão informados pelos usuários, e cuja inferência pode ser difícil.

3.6.1 Um modelo simples de preferências dos usuários

Crawford e Veloso [14] oferecem uma abordagem realista para agendamento em grupo sem assumir que um complexo e abrangente conjunto de dados de treinamento para algoritmos

de inteligência artificial estará disponível. Elas observam que, a menos que o usuário esteja sempre negociando agendamentos com as mesmas pessoas, é difícil construir uma base de conhecimento representativa a partir da qual diferentes comportamentos possam ser recomendados. Elas propõem um modelo de aprendizado muito simples, mas poderoso, que infere as preferências temporais de agendamento somente com base nos eventos agendados previamente.

O mecanismo modela preferências de horário (i.e., se o usuário prefere ter seus compromissos pela manhã ou à tarde) e preferências *back-to-back* (i.e., se o usuário prefere agendar seus eventos em "blocos" ou intercalados por espaços de tempo livre). Ele consiste de um grafo dirigido $P = (S, \vec{E})$ onde:

- S é o conjunto de todos os possíveis estados do calendário do usuário, representado por um vetor de bits no qual cada elemento corresponde a um período de tempo (dia da semana e horário). Um bit 1 indica que há um compromisso nesse dia e horário, e um bit 0 indica que o slot está livre;
- com $s, s' \in S$, temos que $(s, s') \in \vec{E}$ se e somente se é possível obter s' de s com o agendamento de um evento, fazendo com que s' tenha exatamente um bit 1 a mais do que s e os bits de s' tenham valor 1 para todo bit 1 de s; e
- cada aresta $(s, s') \in \vec{E}$ tem um peso correspondente à probabilidade de transitar de s para s'.

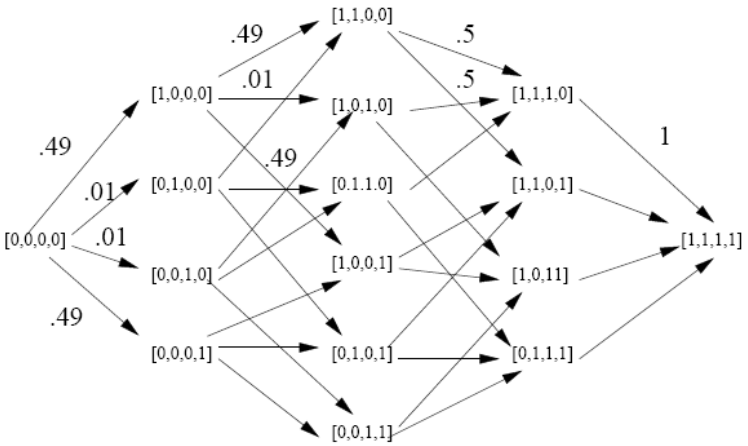


Figura 3.29: Modelo de preferências de agendamento [14]

Para cada evento agendado, o algoritmo de aprendizado incrementa o peso da aresta (s, s') que corresponde aos estados do calendário antes e depois do agendamento. Os agendamentos mais recentes têm um impacto maior neste peso, garantindo que as mudanças de preferências ao longo do tempo sejam capturadas.

Formalmente, a utilidade u do agendamento de um evento para o usuário em um horário t, dado o calendário corrente $cal \in S$, é expressa por:

$$\begin{aligned} u(t, cal) &= \alpha * TOD(t) + \beta * back - to - back - score(t, cal) \\ &+ \gamma * spread - out - score(t, cal) \end{aligned}$$

onde:

- $TOD(t)$ quantifica a preferência do usuário por agendamentos no horário de t, independentemente do estado atual do calendário;
- $back - to - back - score(t, cal)$ vale 2 se o bit correspondente a t em cal tem 1 em ambos os bits adjacentes, 1 se houver o valor 1 em apenas um bit adjacente e 0 caso contrário;
- $spread - out - score(t, cal)$ vale 2 se o bit correspondente a t em cal tem 0 em ambos os bits adjacentes, 1 se houver o valor 0 em apenas um bit adjacente e 0 caso contrário;
- α, β e γ são pesos constantes, com $\beta > 0 \iff \gamma = 0$.

Antes de cada agendamento, o grafo é utilizado para ordenar os horários livres. O nó s que representa o estado corrente do calendário é identificado e os estados sucessores s' tal que $(s, s') \in \vec{E}$ são ordenados da seguinte forma: se o agendamento a partir de s estiver ocorrendo pela primeira vez, os estados s' são ordenados de acordo com suas freqüências; caso contrário, os estados s' são ordenados conforme os pesos das arestas $(s, s') \in \vec{E}$.

Este mecanismo torna-se efetivo rapidamente, com poucos exemplos de aprendizado, e se adapta dinamicamente às modificações de preferências do usuário. Por depender apenas da informação de horário de agendamento e da situação anterior do calendário do usuário, poderia ser incorporado a qualquer sistema de agendamento em grupo.

Em oposição à complexidade de algumas propostas de aplicação de inteligência artificial em agendamento, a eficácia de soluções simples como esta deve ser destacada. No Google Calendar, por exemplo, se o usuário inclui um compromisso "reunião 16h", o sistema é suficientemente inteligente para criar um evento cuja descrição é "reunião", para o qual o horário de início é 16h. Mesmo um mecanismo aparentemente simples como este não é trivial, se complexidades como a interpretação de diferentes linguagens e as diversas formas pelas quais um usuário pode representar a mesma informação forem levadas em conta.

3.6.2 Iniciativas contemporâneas

Atualmente, há dois grandes projetos de pesquisa concentrados no desenvolvimento de ferramentas de agendamento em grupo fortemente baseadas em técnicas de inteligência artificial.

O projeto CMRADAR [54] almeja a construção de um agente de assistência pessoal que aumente a produtividade do usuário ao executar de forma automatizada tarefas comuns de agendamento.

O projeto CALO (Cognitive Assistant that Learns and Organizes), que enfatiza capacidades de processamento de linguagem natural, tem um escopo maior: atuar não somente nas atividades de agendamento, a cargo do módulo PTIME (Personalized Time Manager) [5], mas também em todos os outros aspectos do trabalho em escritórios, incluindo gerência de projetos e organização de documentos.

3.7 Agendamento em grupo como um problema de interfaces: compreendendo as limitações destas abordagens

A experiência trazida pelas primeira e segunda gerações de sistemas de agendamento permite identificar as deficiências das abordagens até agora apresentadas. Os trabalhos de diversos autores (e.g., [15]) têm contribuído para a compreensão das limitações ao desenvolvimento das potencialidades destas importantes aplicações GIM. É possível identificar uma retomada do enfoque dado na década de 80, no qual agendamento é visto como um problema de interfaces e *groupware*.

Mosier e Tammaro [55] destacam o problema do uso parcial, que ocorre quando algumas das pessoas envolvidas em uma negociação de agendamento não fazem uso de um sistema computadorizado. Nestes casos, o processo de negociação perde eficiência, devido à necessidade de adoção dos mecanismos de comunicação "manuais", como telefone e email. Nos anos 80, Ehrlich [29] já notava que a "manutenção de versões eletrônicas"[19] de calendários só seria vantajosa se a maioria das pessoas as utilizassem efetivamente. Há, de fato, uma espécie de círculo vicioso, no qual o baixo nível de utilização das ferramentas de agendamento – em certa medida causado pelas limitações dos protocolos – contribuiu para sua lenta evolução, e vice-versa.

Blandford e Green [7] identificam diversas práticas de agendamento que não são suportadas pelas ferramentas atuais. A impossibilidade de oferecer mais de uma opção de

[19]Este termo denota a idéia de que as agendas eletrônica e em papel deveriam necessariamente coexistir; um paradigma já questionável, especialmente diante da evolução dos dispositivos de computação móvel e redes sem fio.

evento durante processos de negociação pode ser incluída entre tais práticas, já que não conta com suporte dos padrões iCalendar e iTIP.

Outra possível causa para o insucesso de algumas das abordagens passadas seria o fato de que fatores psicológicos decisivos para a utilização de ferramentas de agendamento em um ambiente coletivo ainda não tenham sido totalmente compreendidos. Com relação às propostas de total automatização do processo de agendamento, por exemplo, Ehrlich observava que um agendamento não autorizado de tempo aparentemente livre motivaria nos usuários de um sistema sua total rejeição. Grudin [37] destacava que os calendários eletrônicos não são versões de seus correspondentes em papel, mas artefatos de comunicação cuja utilização têm implicações sociais.

Neste sentido, Palen [61] relata práticas de "agendamento defensivo", através das quais usuários que têm seus horários totalmente disponibilizados em sistemas de agendamento corporativo criam compromissos inexistentes para preencher suas agendas, a fim de não parecerem "desocupados" e evitando que períodos de tempo aparentemente livres sejam alocados para eventos como reuniões. Palen observa, ainda, que as abordagens tradicionais de desenvolvimento de software centradas no indivíduo têm problemas ao serem aplicadas num contexto como este, pois negociações de agendamento em grupo não podem ser avaliadas em um laboratório. Entretanto, sua maior contribuição parece ser a constatação de que o agendamento de eventos é um problema mais de *satisfação* do que de *otimização*, ou seja, um grupo de pessoas tende a agendar um evento tão logo seja encontrado um período de tempo livre comum. Isto reitera a necessidade de possibilitar a adoção de estratégias de anúncio e resposta mais flexíveis para negociação de agendamento com protocolos padrão, para que várias opções de horário (e outros parâmetros de evento) possam ser consideradas simultaneamente.

Como destacam Barták et al. [3], apesar da importância de minimizar as perturbações em um agendamento, é notável a falta de trabalhos de pesquisa neste sentido e o predomínio das abordagens com uma visão unicamente otimizante do problema.

Embora as proposições acadêmicas desta área tenham inicialmente estimulado o desenvolvimento de novas funcionalidades para as aplicações, seu potencial de contribuição foi sendo sub-utilizado pela falta de uma abordagem global de agendamento em grupo. Um exemplo da influência outrora exercida por estas iniciativas é a metáfora visual de transparência proposta por Beard et al. [4], utilizada para mostrar a superposição de eventos simultâneos em sistemas contemporâneos como o Apple iCal.

3.8 Abordagens inovadoras

Quando definimos anteriormente em linhas gerais o problema de agendamento, afirmamos que os recursos e tarefas devem ser escalonados de forma ótima sem dizer, de modo claro, o que é um agendamento ótimo, nem como são especificadas as restrições a serem atendidas.

Na verdade, estabelecer uma métrica de otimalidade não é tarefa trivial. O usuário tem critérios extremamente subjetivos para avaliar a qualidade de um agendamento: enquanto algumas pessoas preferem concentrar seus afazeres no início do dia, outras evitam acordar cedo. Além disso, estes critérios modificam-se ao longo do tempo: para uma pessoa que hoje não tem carro, podem ser necessários intervalos mínimos de uma hora para sua locomoção entre um compromisso e outro, ao passo que daqui a algum tempo, dispondo desse conforto, intervalos tão longos serão desperdício.

Da mesma forma, as restrições que devem ser atendidas, e seu grau de importância, estão sujeitos a variabilidade: assistir a uma peça de teatro é, geralmente, uma tarefa de baixa prioridade; entretanto, se estivermos falando de uma ida ao teatro que já foi desmarcada duas vezes com um amigo, o compromisso passa a ter uma prioridade maior; e, caso se trate de levar um cliente estrangeiro para assistir à última apresentação da peça, temos um compromisso que dificilmente será desmarcado.

É fácil notar que o estabelecimento *a priori* do critério de avaliação da qualidade de um agendamento não é adequado. E, por mais que virtualmente a modelagem explícita e exaustiva dessas restrições seja factível, isso demandaria um volume imenso de informações, o que poderia inclusive violar a privacidade do usuário, e o forçaria a despender mais tempo superespecificando o problema e adequando o sistema às suas necessidades do que aproveitando seu potencial como ferramenta de produtividade.

Embora tragam contribuições relevantes para o entendimento das características do problema com o qual estamos trabalhando, muitas propostas de sistemas de agendamento insistem na tentativa de *resolver o problema independentemente em relação ao usuário*. Porém não se deve negligenciar uma característica intrínseca deste contexto: administrar calendários e agendamentos envolve complexidades e nuances com as quais só o ser humano pode lidar adequadamente.

Em contrapartida, notando a necessidade de ferramentas que *auxiliem o usuário em suas tarefas*, outras abordagens (e.g. [34]) vêm advogando por soluções intermediárias, colaborativas, que conciliem a experiência e a capacidade cognitiva do usuário com os recursos computacionais para manipulação de dados e busca por soluções. Esses sistemas de escalonamento híbrido [38] podem valer-se de duas técnicas para captar informações do usuário: inquiri-lo e observá-lo.

Sistemas que adotam a primeira estratégia dialogam através da interface com o usuário, questionando-o diretamente, oferecendo opções para sua escolha, e tomando decisões a partir das respostas obtidas. Sistemas que observam o usuário utilizam técnicas de inteligência artificial a fim de auxiliá-lo a partir da análise de suas interações com o sistema [86]. Nesse caso, algoritmos de aprendizado alimentam uma base de dados com informações sobre as atividades de agendamento realizadas, a fim de quantificar as preferências dos usuários. Métodos probabilísticos de inferência são então aplicados sobre esses dados para sugerir decisões de agendamento.

3.8.1 LookOut

LookOut, uma extensão para o Microsoft Outlook proposta por Horvitz [39], utiliza ambas as técnicas. O sistema tem uma interface de iniciativa mista, em que tanto o usuário quanto o sistema são agentes, cujas ações são suportadas por inteligência artificial. LookOut analisa o conteúdo de emails trocados entre usuários que tentam marcar um compromisso. Um mecanismo de redes bayesianas é acionado para decidir, a partir do texto de cada email recebido, da agenda do destinatário e de suas interações passadas com o sistema, qual momento seria mais adequado para o agendamento do compromisso e se é conveniente interrompê-lo para fazer esta sugestão. Em caso positivo, o sistema pergunta ao usuário se o agendamento deve ser executado. A figura 3.30 mostra a seqüência de interações do recebimento da mensagem à conclusão do agendamento.

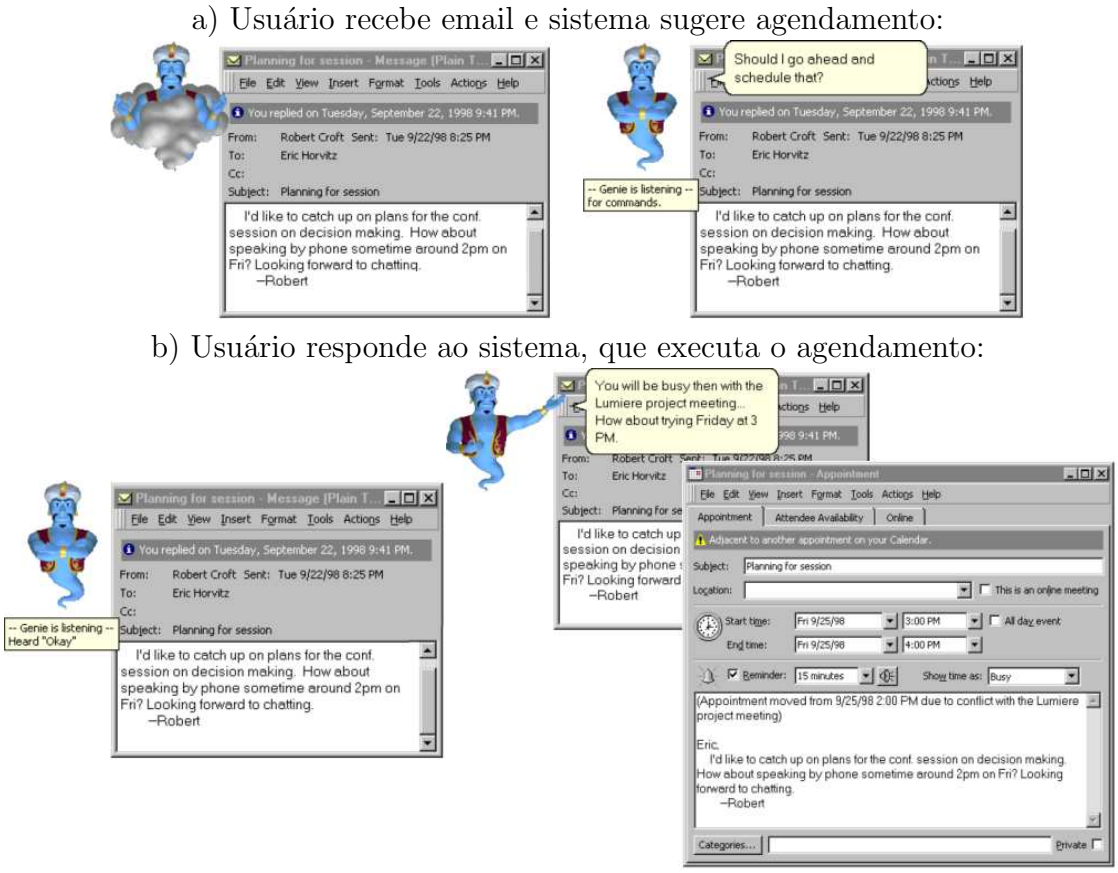


Figura 3.30: Agendamento no sistema LookOut [39]

A respeito de agentes animados em interfaces de iniciativa mista, Swartz [81] observa que, embora potencialmente positiva, sua adoção tem impactos no ambiente de trabalho do usuário que podem determinar o fracasso desta abordagem, utilizando como exemplo a rejeição dos usuários ao *paperclip* do Microsoft Word.

3.8.2 HuGS

Klau et al. [44] desenvolveram o framework Human Guided Search (HuGS), uma solução que envolve o usuário ativamente no processo de otimização. Conforme os usuários participam da busca por soluções ótimas, eles são capazes de contribuir com seus conhecimentos sobre estratégias para resolver problemas reais, além de compreender e justificar melhor as respostas do sistema. Além disso, a experiência do usuário pode guiar a busca pela otimalidade num espaço de soluções e ajudar o sistema a sair de máximos locais, o que resulta em uma convergência mais rápida para soluções melhores.

Os seguintes conceitos são fundamentais neste framework:

- um **problema** é uma instância do problema de otimização em questão. Por exemplo, o exemplo 2 da seção 3.5.1, que é uma instância de TR, é um problema;
- uma **solução** é uma resposta – factível ou não – para um problema. Por exemplo, um agendamento com o $n+1$-ésimo evento é uma solução para TR. O agendamento $T(6)$ mostrado na figura 3.23 é uma solução para o problema exemplificado no item acima;
- **scores** permitem que duas soluções quaisquer sejam comparadas. Em um problema jobshop, por exemplo, um agendamento pode ser considerado melhor que outro se o seu makespan for menor;
- os **movimentos** correspondem à função sucessor do processo de busca, e geram modificações que podem ser aplicadas a uma solução para gerar outras. Por exemplo, as operações SR, SL, CP, XP, CSL e CSR utilizadas na busca A* para TR são movimentos; e
- um número finito de **elementos** que fazem parte do problema, e que podem ser alterados pelos movimentos. Para problemas jobshop, por exemplo, os elementos são as tarefas, que podem ser agendadas e movidas de uma máquina para outra.

A partir de uma **solução inicial**, algoritmos que implementam diferentes estratégias de busca são invocados. Eles executam os movimentos que geram os próximos nós a serem expandidos no grafo de busca. Um mecanismo de **mobilidades** (alta, média e baixa) controla quais elementos podem ser modificados neste processo. A cada iteração, um elemento com mobilidade alta é escolhido para o próximo movimento. Este movimento pode envolver outros elementos, desde que eles não tenham mobilidade baixa. Operando com as mobilidades, o usuário contribui com seu conhecimento do problema para o processo de solução ao mesmo tempo em que torna o processo de busca mais eficiente, indicando que a busca deve ser focada em movimentos com os elementos de alta mobilidade, e diminuindo o espaço de soluções possíveis ao fixar elementos como tendo mobilidade baixa.

O framework permite que os usuários tentem melhorar a solução corrente agindo diretamente sobre o processo de busca:

1. alterando as mobilidades dos elementos e escolhendo manualmente um movimento a ser aplicado à solução corrente;

2. invocando, monitorando, e interrompendo um processo de busca por uma melhor solução; e

3. voltando a uma solução obtida previamente.

Esta operação é realizada através de uma camada de **visualização**. Ela faz uso de metáforas visuais que permitem ao usuário controlar o processo de otimização, ora observando o progresso do algoritmo de busca, ora modificando diretamente a solução.

O framework HuGS é implementado em uma biblioteca Java para otimização interativa, e foi aplicado a diversos problemas combinatoriais de alta complexidade, como desenho de grafos e seqüenciamento de proteínas. Algoritmos de busca por aprofundamento iterativo, busca gulosa e busca tabu estão disponíveis para utilização.

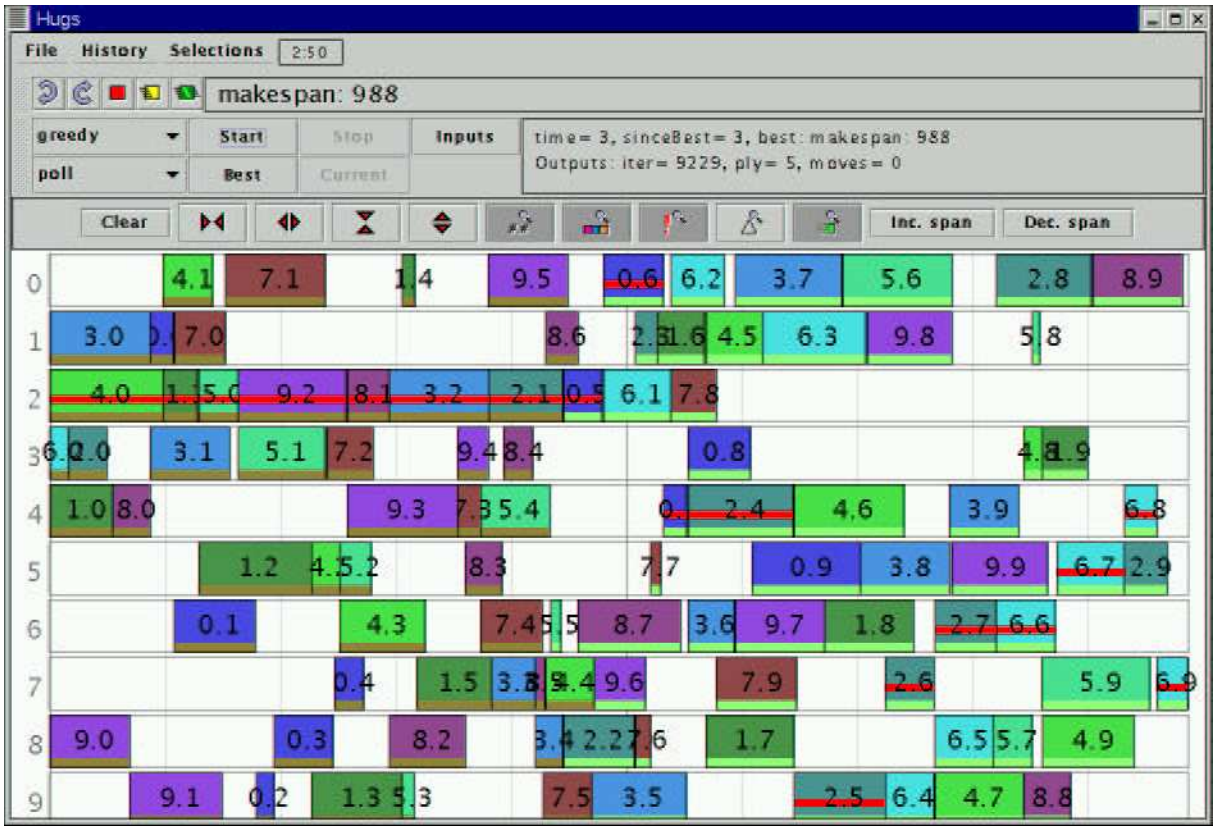


Figura 3.31: Aplicação interativa HuGS para jobshop [48]

A implementação de busca tabu [43] foi utilizada para construir uma aplicação interativa para uma versão de jobshop na qual cada tarefa é composta de diversas operações que devem ser executadas obedecendo uma certa ordem, conforme mostra a figura 3.31.

Na busca tabu, a mobilidade de um elemento é reduzida conforme a realização de movimentos nos quais ele está envolvido. A lista de movimentos tabu, portanto, é representada pelo conjunto de elementos com mobilidade baixa.

O painel superior da interface permite ao usuário controlar a busca, selecionando o algoritmo a ser utilizado e agindo sobre ele. Os botões que implementam uma metáfora baseada em semáforos permitem ao usuário atribuir mobilidades alta (verde), média (amarelo) e baixa (vermelho) aos elementos selecionados.

User Hints Um framework similar a HuGS foi proposto por Nascimento e Eades [24]. Além de ajudar o usuário a buscar por soluções ótimas, User Hints oferece mecanismos para definir o próprio problema de otimização incrementalmente, ao longo do processo de busca, através da incorporação de novas restrições em tempo de execução.

Capítulo 4

Uma nova abordagem

Propomos uma nova abordagem para agendamento em grupo que consiste em entendê-lo como um problema multidisciplinar, introduzir novos conceitos que possibilitem o projeto de ferramentas mais poderosas, padronizar as extensões de protocolos utilizadas para implementar as novas funcionalidades, e desenvolver aplicações que provem a viabilidade destes caminhos, fomentando o surgimento dos sistemas de terceira geração.

4.1 Agendamento em grupo, um desafio multidisciplinar

As propostas já apresentadas para agendamento em grupo falham ao não reconhecer a necessidade de abordar de forma integrada as diferentes disciplinas relevantes para um problema desta complexidade, que são ilustradas na figura 4.1.

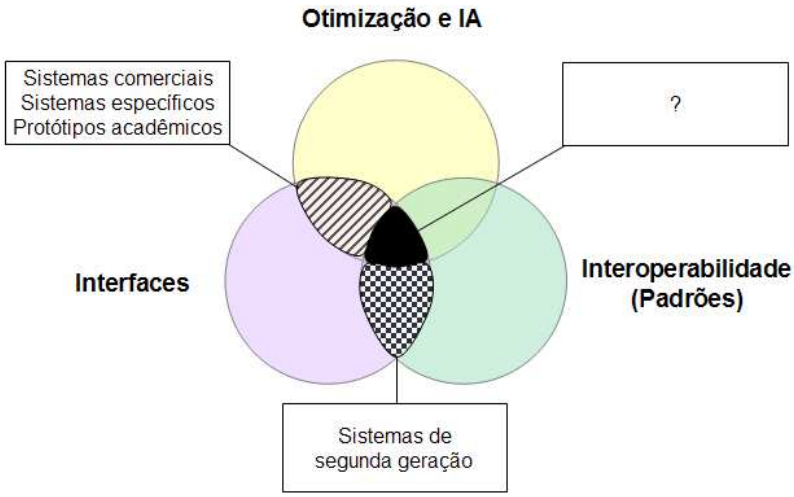


Figura 4.1: Intersecção de áreas de conhecimento nas soluções para agendamento

Destacamos a intersecção entre os campos de otimização e inteligência artificial com interfaces, onde encontram-se as soluções descritas nas seções 3.5 e 3.6, e entre interfaces e interoperabilidade com padrões abertos, correspondente às ferramentas de segunda geração.

O êxito das abordagens inovadoras apresentadas na seção 3.8 indica o potencial de uma visão articulada e multidisciplinar. Não temos conhecimento de nenhuma abordagem prévia que proponha a combinação de todos estes elementos para oferecer soluções de agendamento em grupo: técnicas de otimização e inteligência artificial, interfaces de iniciativa mista, e suporte aos protocolos padrão de compartilhamento de calendários e agendamento para comunicação interoperável.

Em nossa opinião, estas áreas guiarão o desenvolvimento da próxima geração de sistemas de agendamento por meio de uma promissora simbiose. Acreditamos que nenhuma área de pesquisa deva ser privilegiada em detrimento de outras. De fato, identificamos nas abordagens mais bem sucedidas até o momento enfoques que não são exclusivos a um único aspecto de agendamento em grupo. Ao contrário de abordagens isoladas, portanto, o objetivo de futuros trabalhos deve estar na proposição de soluções integradas para problemas de agendamento do ponto de vista do usuário, e no desenvolvimento de aplicações utilizando resultados de diversas disciplinas.

4.2 Introduzindo novos conceitos

Na medida em que o protocolo CalDAV tiver sucesso e, tornando-se amplamente utilizado, possibilitar de forma transparente trocas de dados para compartilhamento de calendários e agendamento entre aplicações, plataformas e dispositivos heterogêneos, a demanda por sistemas de agendamento em grupo mais poderosos irá, certamente, motivar o desenvolvimento de novos conceitos que incorporem e modelem seus requisitos.

A partir de um diagnóstico de situações usuais que não são adequadamente expressas nos sistemas de agendamento atuais, alguns destes conceitos são apresentados.

4.2.1 Tarefas líquidas

Comumente, as pessoas estão a cargo de tarefas de médio e longo prazo que consomem volumes consideráveis de tempo, e têm um prazo final para conclusão. Tipicamente, há também uma estimativa a respeito do tempo total necessário de dedicação a estas tarefas, que pode ser uma estimativa inicial refinada ao longo do tempo. Devido à natureza da tarefa, pode haver limites nos períodos mínimo e máximo de tempo dedicado. Tarefas mecânicas que requerem pouco esforço intelectual e nenhuma preparação especial podem ser executadas mesmo em períodos curtos, enquanto que outras podem apresentar

um *start-up time* não negligenciável, exigindo um período mínimo de concentração sem interrupções, como escrever um artigo, por exemplo.

Chamamos este tipo de atividade de médio ou longo prazo de **tarefa líquida**, pois ela "preenche" períodos de tempo livre até sua conclusão. As tarefas líquidas são continuamente alocadas em slots futuros que satisfaçam às suas restrições. Tratando-se de atividades mais extensas, é irreal fazer uma pré-alocação completa em eventos, pois certamente imprevistos farão com que períodos reservados para uma tarefa líquida dêem espaço a eventos de maior prioridade, adiando a realização da tarefa. A alocação de tarefas líquidas tende a ser mais "sólida" no curto prazo, quando há menos incertezas com relação aos eventos que garantirão sua realização, e mais "fluida" num futuro mais distante.

Embora seja possível argumentar que uma tarefa líquida é um to-do, dado que sua semântica permite as funcionalidades de registro e acompanhamento de progresso implementadas pelas aplicações mais populares, tarefas líquidas têm importantes características que os to-dos não têm:

- **granularidade mínima**: o requisito mínimo de tempo contínuo para a execução da tarefa, que também inclui o tempo de *start-up*;
- **granularidade máxima**: o limite de tempo contínuo que pode ser utilizado para a execução da tarefa; e
- **tempo restante para finalização**: uma característica dinâmica que é atualizada ao longo do tempo, e tende a se tornar mais precisa à medida em que o prazo final para conclusão da tarefa se aproxima.

Devido ao caráter difuso e preemptível das tarefas líquidas, que as sujeitam a sucessivos adiamentos em favor de eventos mais urgentes, sua alocação de tempo pode ser facilmente subestimada. Uma aplicação de agendamento poderia monitorar as modificações na alocação de uma tarefa líquida em eventos ao longo do tempo e, com base nesse histórico, oferecer ao usuário previsões de situações críticas que podem ocorrer no futuro.

As tarefas líquidas podem, ainda, ser a ligação – hoje inexistente – entre ferramentas de agendamento e sistemas de planejamento, tais como aqueles que trabalham sob a perspectiva de gerência de projetos: as fases de projeto podem ser melhor representadas como tarefas líquidas que, por sua vez, seriam agendadas como vários eventos para os participantes do projeto a cargo de sua execução. A partir do registro destes eventos, as ferramentas de gerência de projetos poderiam automaticamente obter informações para replanejar as tarefas pendentes com base no trabalho já realizado e gerar relatórios previsto *versus* realizado.

4.2.2 Eventos imprecisos e eventos alternativos

Na verdade, a noção de tarefa líquida está contida em um conceito mais abrangente, o de **eventos imprecisos**. Um evento impreciso é um esboço de um evento (ou conjunto de eventos) ainda não agendado, para o qual os horários de início e término não estão definidos. São aplicáveis a eventos imprecisos, portanto, todas as propriedades pertinentes às tarefas líquidas.

Em um evento impreciso, os períodos de tempo disponíveis de um participante para o agendamento real do evento devem ser representados. Além disso, estes períodos podem ser ordenados, representando a conveniência de sua utilização para o evento em questão. Uma reunião de 1 hora de duração ocorrendo – preferencialmente – na segunda-feira entre 8h e 11h, ou na quinta-feira às 14h, é um exemplo de um evento impreciso.

A capacidade de representar um evento imprecisamente é especialmente importante durante negociações de agendamento, quando os participantes devem entrar em consenso a respeito de uma descrição de evento que inicialmente não é exata. Especificações de evento inexatas são apropriadas para refinamento durante o processo de negociação, até que o consenso seja atingido e uma completa descrição do evento agendado seja estabelecida.

Algumas situações podem, adicionalmente, requerer múltiplas descrições alternativas (imprecisas ou não) de um evento. Estes **eventos alternativos** podem ser utilizados durante negociações de agendamento para oferecer múltiplas opções de eventos com mais de uma alternativa para parâmetros que não os horários de início, dentre as quais os participantes possam escolher a mais adequada. Também deve ser possível ordenar qualitativamente estes eventos alternativos, abrindo espaço para que as preferências dos usuários sejam incluídas no processo de negociação.

O evento impreciso exemplificado há pouco poderia ser estendido e representado como duas descrições de eventos que são alternativas: uma reunião de 1 hora de duração ocorrendo – preferencialmente – no escritório A na segunda-feira entre 8h e 11h, ou no escritório B na quinta-feira às 14h ou na sexta após 16h30min.

Como mostrado na seção 2.5, múltiplas descrições alternativas e períodos de tempo para a realização de um evento podem ajudar um grupo de participantes a convergir mais rapidamente para uma descrição aceita por todos [71]. Embora na maioria das situações isto signifique alcançar consenso a respeito de um horário que satisfaça um período de tempo livre em comum, outras propriedades de um evento também podem estar em negociação (e.g., `LOCATION`, `DURATION`, `ATTENDEE`s), o que leva ao conceito de eventos alternativos[1].

[1]Eventos imprecisos e alternativos ainda refletem mais adequadamente as disponibilidades e preferências com respeito ao tempo dos participantes, pois não se limitam a uma visão binária na qual períodos são vistos como livres ou ocupados através de consultas que retornam componentes `VFREEBUSY`. Com estes conceitos, é possível representar os períodos disponibilizados para agendamento exclusivamente no curso de uma dada negociação, que podem depender inclusive das demais propriedades do evento.

4.3 Especificação de extensões de protocolos

Embora a necessidade de interfaces de usuário colaborativas para viabilizar a utilização de otimização e inteligência artificial em atividades de agendamento já seja bem compreendida, estas abordagens têm praticamente ignorado a necessidade de protocolos padronizados para o transporte de dados nas negociações de agendamento. Soluções práticas que, ao mesmo tempo, administrem e minimizem as limitações dos protocolos existentes – explicitadas na seção 3.4 – são urgentes.

A família de protocolos iCalendar representa mais de uma década de trabalhos no desenvolvimento de padrões com o objetivo de viabilizar a comunicação interoperável entre diferentes sistemas. O desenvolvimento de ferramentas de agendamento em grupo mais flexíveis, inteligentes e poderosas demandará, inevitavelmente, novos requisitos de protocolos para suportar funcionalidades adicionais. No projeto de futuros sistemas, extensões para os protocolos existentes podem ser consideradas. Se implementadas, sua especificação através de padrões abertos é a única forma de garantir a manutenção das condições de interoperabilidade.

4.3.1 Eventos imprecisos e alternativos para iCalendar e iTIP

O formato iCalendar não oferece nenhuma forma de descrever um evento imprecisamente, ou como representar múltiplas alternativas de um evento. Neste último caso, uma série de mensagens iTIP também não pode ser utilizada, pois a especificação estabelece que a última mensagem recebida com maior valor `SEQUENCE` torna todas as anteriores obsoletas.

Com a finalidade de flexibilizar estes padrões – minimizando suas limitações – e tornar as negociações de agendamento com iTIP mais eficientes, propomos as seguintes extensões:

1. novos componentes de calendário `VIMPRECISEEVENT` e `VALTERNATIVEEVENTS`, para a representação de eventos imprecisos e alternativos;

2. novas propriedades `MIN-GRANULARITY` e `MAX-GRANULARITY`, que permitem a utilização do componente `VIMPRECISEEVENT` para a representação de tarefas líquidas, com suas granularidades mínima e máxima;

3. novos parâmetro e propriedade `RANK`, que podem ser utilizados para expressar preferências com relação a eventos alternativos e aos períodos de tempo associados a eventos imprecisos; e

4. regras iTIP adicionais para a utilização dos novos componentes de calendário.

Seu propósito é viabilizar uma representação de eventos mais flexível e a utilização de estratégias de negociação de agendamento mais eficientes, tratando de forma realística

as questões de compatibilidade com aplicações já existentes. As três primeiras extensões adicionam elementos ao padrão iCalendar, enquanto a última possibilita a utilização de `VIMPRECISEEVENT` e `VALTERNATIVEEVENTS` com iTIP.

O componente VIMPRECISEEVENT

O componente de calendário `VIMPRECISEEVENT`, utilizado para representar eventos imprecisos, consiste, em termos práticos, de um componente `VEVENT` modificado. Propriedades como `DTSTART` e `DTEND`, presentes nos componentes de eventos tradicionais, são excluídos, pois não fazem sentido em uma descrição de evento "não-determinística".

Diversos períodos de tempo passíveis de agendamento podem ser associados a um evento impreciso. Isto é feito através da inclusão de componentes `VFREEBUSY`, que já são parte do padrão iCalendar, e de componentes `VAVAILABILITY`, propostos em um draft de Daboo e Desruisseaux [18], que oferecem meios para representar intervalos recorrentes de tempo livre.

Os horários disponibilizados para agendamento são constituídos pela união dos períodos especificados nos componentes `VFREEBUSY` e `VAVAILABILITY` do evento impreciso. As propriedades `FREEBUSY` dos componentes `VFREEBUSY` não podem especificar períodos de tempo que tenham intersecção, mas podem coincidir com intervalos expressos pelos componentes `VAVAILABILITY`, sobrescrevendo-os.

```
BEGIN:VIMPRECISEEVENT
UID:20071005T133225Z-00001@example.com
DTSTAMP:20071005T133225Z
SUMMARY:Reunião de departamento
DURATION:PT2H
BEGIN:VFREEBUSY
UID:20071005T133225Z-00001A@example.com
DTSTAMP:20071005T133225Z
FREEBUSY;FBTYPE=FREE:20080401T080000Z/20080401T120000Z
END:VFREEBUSY
END:VIMPRECISEEVENT
```

Figura 4.2: Exemplo de evento impreciso simples

A figura 4.2 mostra um exemplo de evento impreciso simples, com duração de 2h, que pode ocorrer no intervalo descrito pelo componente `VFREEBUSY`, entre 8h e 12h do dia 01/04/2008.

A figura 4.3 mostra um exemplo de um evento impreciso com recorrência mensal, e diversas opções de horário representadas pelas propriedades `FREEBUSY` dos componentes

```
BEGIN:VIMPRECISEEVENT
UID:20071005T133225Z-00001@example.com
DTSTAMP:20071005T133225Z
SUMMARY:Eleição do novo síndico
RRULE:FREQ=MONTHLY;BYDAY=MO,TU,WE,TH,FR
BEGIN:VFREEBUSY
UID:20071005T133225Z-00001-A@example.com
DTSTAMP:20071005T133225Z
FREEBUSY;FBTYPE=FREE;RANK=100:20080328T150000Z/PT3H
FREEBUSY;FBTYPE=FREE;RANK=100:20080329T163000Z/PT1H30M
FREEBUSY;FBTYPE=FREE;RANK=80:20080330T140000Z/PT6H
END:VFREEBUSY
BEGIN:VFREEBUSY
UID:20071005T133225Z-00001-B@example.com
DTSTAMP:20071005T133225Z
COMMENT:Se necessário, posso tentar reagendar
FREEBUSY;FBTYPE=BUSY-TENTATIVE;RANK=10:20080401T160000Z/PT1H
END:VFREEBUSY
END:VIMPRECISEEVENT
```

Figura 4.3: Exemplo de evento impreciso recorrente com vários períodos de tempo

`VFREEBUSY` incluídos, cujas preferências são expressas através do parâmetro `RANK`. Em um evento impreciso, a propriedade `RRULE` não descreve uma regra de recorrência para o cálculo das instâncias exatas do evento, mas denota a freqüência que o evento agendado poderia ter.

Note que o uso do valor `BUSY-TENTATIVE` para o parâmetro `FBTYPE` referente ao intervalo entre 16h e 17h de 01/04/2008 denota que, embora haja eventos provisoriamente agendados neste período, ele pode ser considerado para agendamento. Este último intervalo foi representado na propriedade `FREEBUSY` de um componente `VFREEBUSY` distinto, de forma que o conteúdo da propriedade `COMMENT` refere-se somente a ele.

A figura 4.4 mostra um exemplo de um evento impreciso representando uma tarefa líquida que descreve um tratamento dental com duração total de 10 horas, das quais 20% já foram completadas. As propriedades `MIN-GRANULARITY` e `MAX-GRANULARITY` indicam que as 8 horas restantes podem ser agendadas em consultas de pelo menos 1 hora e não mais do que 2 horas de duração.

O componente `VAVAILABILITY` especifica uma disponibilidade recorrente das 16h às 18h, em todas as terças-feiras e quintas-feiras entre os dias 15/03/2008 e 15/04/2008. O componente `VFREEBUSY`, além de sobrescrever a disponibilidade expressa pelo componente

`VAVAILABILITY` entre 16h e 17h de 01/04/2008, especifica um intervalo adicional de tempo livre entre 18h e 19h em 03/04/2008. Note o uso da propriedade `RANK` no sub-componente `AVAILABLE` de `VAVAILABILITY`.

```
BEGIN:VIMPRECISEEVENT
UID:20071005T133225Z-00001@example.com
DTSTAMP:20071005T133225Z
SUMMARY:Dentista
DURATION:PT10H
PERCENT-COMPLETE:20
MIN-GRANULARITY:PT1H
MAX-GRANULARITY:PT2H
BEGIN:VAVAILABILITY
UID:20071005T133225Z-00001-A@example.com
DTSTAMP:20071005T133225Z
DTSTART:20080315T160000Z
BEGIN:AVAILABLE
UID:20071005T133225Z-00001-A1@example.com
RANK:95
DTSTART:20080315T160000Z
DTEND:20080315T180000Z
RRULE:FREQ=WEEKLY;BYDAY=TU,TH
END:AVAILABLE
END:VAVAILABILITY
BEGIN:VFREEBUSY
UID:20071005T133225Z-00001-B@example.com
DTSTAMP:20071005T133225Z
FREEBUSY;FBTYPE=BUSY-TENTATIVE;RANK=33:20080401T160000Z/PT1H
FREEBUSY;FBTYPE=FREE;RANK=60:20080403T180000Z/PT1H
END:VFREEBUSY
END:VIMPRECISEEVENT
```

Figura 4.4: Exemplo de evento impreciso representando tarefa líquida

O componente VALTERNATIVEEVENTS

Os componentes de calendário `VALTERNATIVEEVENTS`, utilizados para representar eventos alternativos, são coleções de componentes de eventos "regulares" (`VEVENT`) e imprecisos (`VIMPRECISEEVENT`) que contêm propriedades comuns a todas as alternativas.

A figura 4.5 ilustra um exemplo de eventos alternativos. São colocadas como opções mutuamente exclusivas as descrições dadas pelo componente `VEVENT` e pelo componente

VIMPRECISEEVENT incluídos. Note o uso da propriedade RANK para indicar as preferências pelos eventos regular e impreciso.

```
BEGIN:VALTERNATIVEEVENTS
UID:20071005T133225Z-00001@example.com
DTSTAMP:20071005T133225Z
SUMMARY:Reunião de departamento
COMMENT:Tentando as opções mais prováveis...
BEGIN:VEVENT
UID:20071005T133225Z-00001-A@example.com
DTSTAMP:20071005T133225Z
DTSTART:20080406T090000Z
DTEND:20080405T120000Z
RANK:100
COMMENT:Seria perfeito se nos encontrássemos no mesmo horário!
END:VEVENT
BEGIN:VIMPRECISEEVENT
UID:20071005T133225Z-00001-B@example.com
DTSTAMP:20071005T133225Z
RANK:70
BEGIN:VFREEBUSY
UID:20071005T133225Z-00001-B-1@example.com
DTSTAMP:20071005T133225Z
FREEBUSY;FBTYPE=FREE;RANK=100:20080329T163000Z/PT1H30M
FREEBUSY;FBTYPE=FREE;RANK=80:20080330T140000Z/PT6H
FREEBUSY;FBTYPE=BUSY-TENTATIVE;RANK=33:20080401T160000Z/PT1H
FREEBUSY;FBTYPE=FREE;RANK=60:20080403T180000Z/PT1H
END:VFREEBUSY
END:VIMPRECISEEVENT
END:VALTERNATIVEEVENTS
```

Figura 4.5: Exemplo de eventos alternativos

Extensão para iTIP

Para que os eventos imprecisos e alternativos possam ser utilizados em negociações de agendamento, os métodos iTIP devem ser aplicados aos novos componentes VIMPRECISEEVENT e VALTERNATIVEEVENTS. Com exceção do método ADD, destinado a agregar instâncias específicas a eventos recorrentes, todos os demais métodos iTIP são aplicáveis a estes componentes, e têm comportamento análogo àquele descrito para o componente VEVENT na subseção 3.2.2.

Adicionalmente, os três componentes para eventos regulares, imprecisos e alternativos (respectivamente, `VEVENT`, `VIMPRECISEEVENT` e `VALTERNATIVEEVENTS`) passam a ser utilizados indistintamente uns com os outros durante uma negociação de agendamento. Por exemplo, um participante que recebesse um convite para o evento impreciso mostrado no exemplo da figura 4.2 através do método `REQUEST` poderia fazer uma contra-proposta sugerindo uma descrição com os eventos alternativos exemplificados na figura 4.5 através do método `COUNTER`. A utilização do mesmo valor na propriedade `UID` garante a identificação do componente de evento como parte de uma negociação em curso.

Limitações destas extensões

Do modo como concebemos as extensões para iCalendar, os eventos imprecisos não são flexíveis para a representação de parâmetros que não o horário do evento. Por exemplo, um evento que pudesse ocorrer em dois locais exigiria a utilização de um evento alternativo como duas descrições de eventos diferindo apenas pela propriedade `LOCATION`. Embora dessa limitação possam resultar em descrições verbosas de eventos com mais de uma alternativa de propriedades não-temporais, isto não chega a ser um problema, mesmo porque tipicamente a propriedade de interesse em uma negociação é o seu horário de início.

Com relação à representação de eventos imprecisos, diferentemente do que sugerem Sayers e Letsinger [67], entendemos uma descrição do tipo "Gostaria de encontrar-me com você em Campinas na terça-feira pela manhã, ou em São Paulo na sexta-feira à tarde" não deva ser admitida. Para máxima interoperabilidade os períodos de tempo devem ser descritos "deterministicamente", de forma a não depender de mecanismos de inteligência artificial e/ou sutilezas culturais para sua compreensão. Em um evento impreciso, portanto, esta requisição seria descrita com um componente `VALTERNATIVEEVENTS` com dois eventos imprecisos, um correspondente à opção (Campinas, terça-feira, 8h – 12h) e outro correspondente a (São Paulo, sexta-feira, 14h30min – 18h).

A possibilidade de expressar relações semânticas mais poderosas, contudo, deve ser analisada. Uma relação disjuntiva do tipo `one-of`(location$_1$, ..., location$_k$), por exemplo, poderia substituir com muita eficiência um componente com k alternativas de eventos. Porém, mais do que ser tratada em uma proposta de extensão dos formatos atuais, esta é uma questão inerente ao padrão iCalendar. Acreditamos que a consolidação do protocolo CalDAV e as demandas por flexibilidade poderão motivar uma mudança de paradigma, com a evolução das iniciativas xCal ou xCalendar para um padrão baseado em XML que não só transponha a ontologia do padrão iCalendar para um outro formato, mas tire proveito do poder de representação hierárquico-recursivo natural de esquemas XML.

Observamos, ainda, que a necessidade de tratar intercambiavelmente três componentes de eventos aumenta a complexidade da implementação do fluxo de negociações de agen-

damento; dificuldade que também poderá ser superada pela adoção futura de um padrão mais flexível de representação de dados que permita a representação destas três categorias como casos particulares de um conceito mais abrangente de eventos.

4.3.2 Draft de especificação das extensões

De modo a compreender melhor o andamento dos trabalhos de desenvolvimento e evolução dos protocolos, participamos das listas de discussão nas quais são debatidas as atividades do grupo Calsify[2] e a especificação do protocolo CalDAV[3], bem como de algumas das conferências virtuais promovidas pelo grupo Calsify[4].

Submetemos, então, ao diretório de Internet-Drafts da IETF, um draft de especificação destas propostas de extensões para iCalendar e iTIP [74]. O detalhamento formal das extensões encontra-se no apêndice A, que contém a íntegra do conteúdo do draft. Conforme sugestão dos desenvolvedores de protocolos, esta especificação foi colocada para debate na lista de discussão dos membros do grupo Calsify.

Recebemos um modesto número de opiniões a respeito da proposta, algumas bastante animadoras. Em geral, os comentários ressaltam a pertinência de buscar a padronização das extensões no contexto do processo de desenvolvimento de protocolos da IETF.

As principais ressalvas ficam por conta das dificuldades de interoperabilidade que a criação de novos componentes poderia trazer. Foram feitas algumas sugestões sobre como evitar maiores modificações, adaptando os componentes já existentes do padrão iCalendar para representar eventos imprecisos e alternativos. Para fornecer mais elementos para a discussão, introduzimos algumas considerações a respeito dos benefícios e limitações desta abordagem mais conservadora no apêndice do próprio draft.

Também percebemos, entretanto, que parte da comunidade ainda considera tais propostas muito acadêmicas[5], e acredita que o email seja uma ferramenta suficiente para a negociação de múltiplas (contra-)propostas de agendamento. Uma possível razão para tal percepção seria o fato de que as propostas têm origem em um trabalho de pós-graduação, quando quase toda a comunidade é formada por desenvolvedores atuantes nas maiores empresas de desenvolvimento de software, o que poderia denotar uma abordagem muito teórica e distante das dificuldades reais de implementação.

Felizmente, mensagens adicionais postadas recentemente na lista estão dando continuidade às discussões, e alguns membros da comunidade sugerem caminhos para implementar as extensões de forma a maximizar sua compatibilidade com as aplicações existentes, indicando que a flexibilização dos padrões é vista como um tema relevante. Consideramos

[2] `http://lists.osafoundation.org/mailman/listinfo/ietf-calsify`.
[3] `http://lists.osafoundation.org/mailman/listinfo/ietf-caldav`.
[4] `http://www.ietf.org/meetings/ietf-logs/calsify`.
[5] `http://www.ietf.org/meetings/ietf-logs/calsify/2007-11-07.html`.

que o objetivo principal do draft, de expor as limitações dos padrões e motivar reflexões a respeito de como enfrentá-las, foi atingido.

O amadurecimento do trabalho nesta especificação poderá culminar com sua publicação como uma RFC experimental, o que ainda depende de consenso sobre sua relevância e conveniência na lista de discussão e com relação aos próprios membros do CalConnect. Um outro caminho possível seria empreender uma discussão mais ampla e unificada, sobre maior flexibilidade para os padrões, o que também contemplaria uma análise aprofundada das propostas de representação XML para iCalendar. De todo modo, o encaminhamento dessas iniciativas deve ser feito gradualmente, de forma a não concorrer com a necessária finalização dos trabalhos do grupo Calsify com respeito às versões "bis" dos padrões já existentes.

4.4 Esboçando os sistemas de terceira geração

Uma terceira geração de sistemas de agendamento, mais inteligente e mais poderosa, deverá se originar do uso dos novos conceitos resultantes desta abordagem multidisciplinar. Nesta nova geração de aplicativos, as atividades de agendamento em grupo não só receberão um tratamento diferenciado com relação aos sistemas atuais, mas serão um paradigma fundamental.

Acreditamos que a aplicação efetiva desta nova abordagem será completa somente com o desenvolvimento de ferramentas usáveis que possam se inserir em ambientes de trabalho em grupo. Portanto, futuros trabalhos de pesquisa devem considerar a prototipação de soluções de software que demonstrem de forma prática a aplicabilidade da adoção de suas propostas. Isto permitirá um diálogo mais direto com a comunidade de usuários e desenvolvedores, e estimulará a implementação de inovações nos produtos mais populares das grandes companhias.

Em última instância, a consolidação dessa terceira geração de sistemas ocorrerá na medida em que tais aplicações se tornarem amplamente disseminadas. Para tanto, propomos uma ferramenta destinada à prova de conceito que incorpora estas abordagens promissoras: iScheduler.

Capítulo 5

iScheduler

Neste capítulo, introduzimos o iScheduler, um agendador em grupo *i*nteligente, *i*nterativo e *i*nteroperável (baseado nos protocolos da família *i*Calendar e nas extensões propostas no capítulo 4). Seu propósito é demonstrar, de forma prática, a efetividade das abordagens apresentadas para o desenvolvimento de novos sistemas de agendamento em grupo. Estamos especialmente interessados em exercitar a aplicação de mecanismos de otimização e inteligência artificial na geração de agendamentos mais eficientes, enquanto preservando total aderência aos protocolos de comunicação padrão.

São apresentados a arquitetura conceitual do iScheduler, uma sugestão para sua implementação, a descrição dos protótipos implementados a título de prova de conceito, e os comentários a respeito dos resultados obtidos.

5.1 Arquitetura

A figura 5.1 ilustra a arquitetura do iScheduler. Basicamente, há uma divisão em quatro módulos: interface, núcleo de armazenamento de informações de calendário, algoritmos centrais de otimização e inteligência artificial – não presentes nos sistemas de segunda geração – e funcionalidades de intercomunicação.

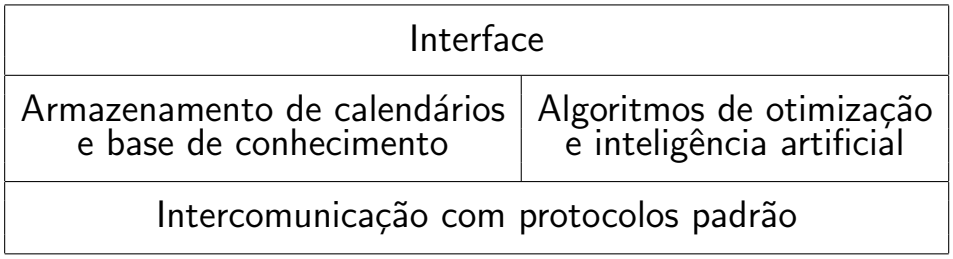


Figura 5.1: Arquitetura do iScheduler

A camada de interface permite ao usuário tanto manipular seus calendários, da mesma

forma que nas aplicações tradicionais, quanto interagir com os algoritmos de otimização e inteligência artificial através de mecanismos de iniciativa mista. Juntamente com as informações de calendários é mantida uma base de conhecimento sobre os agendamentos realizados, que alimenta e é alimentada pelos algoritmos centrais da arquitetura. A comunicação com outras aplicações é feita por uma camada que implementa os protocolos padrão, garantindo interoperabilidade.

5.2 Implementações

Como prova de conceito da nova abordagem que propomos e da factibilidade de implementação do iScheduler, foram realizadas, de forma independente, as seguintes implementações:

1. uma biblioteca de entrada e saída para dados no padrão iCalendar com suporte aos novos componentes `VIMPRECISEEVENT` e `VALTERNATIVEEVENTS`; e

2. um algoritmo interativo para Timetable Rearrangement.

Enquanto que a primeira implementação diz respeito tanto ao armazenamento de informações de calendários quanto à intercomunicação com outras aplicações, já que ambas dependem do formato padrão iCalendar, a segunda envolve a interação da camada de interface com um algoritmo de agendamento. Não há, entretanto, integração funcional entre estas implementações.

5.2.1 Biblioteca para extensões do padrão iCalendar

Uma extensão da biblioteca Java iCal4j[1], que implementa funcionalidades de entrada e saída para todos os componentes especificados na RFC 2445, foi realizada. A biblioteca iCal4j converte dados no formato iCalendar para objetos Java, e vice-versa. A extensão da biblioteca que foi implementada dá suporte aos novos componentes para eventos imprecisos e alternativos.

Para o componente `VIMPRECISEEVENT`, apenas períodos de tempo expressos via componentes `VFREEBUSY` são suportados. Optamos por uma implementação parcial devido à complexidade do componente `VAVAILABILITY`, que deverá ser incorporado à própria biblioteca iCal4j em breve[2].

A seção 1 do apêndice B traz detalhes técnicos da estrutura da biblioteca iCal4j, e de como ela foi estendida para suportar os novos componentes.

[1] `http://ical4j.sourceforge.net/`.

[2] `http://sourceforge.net/forum/forum.php?thread_id=1760635&forum_id=368290`.

5.2.2 Algoritmo interativo para Timetable Rearrangement

Com base no código da aplicação para jobshop baseada na biblioteca do framework HuGS[3], um algoritmo interativo para TR foi desenvolvido. As estratégias apresentadas a seguir foram utilizadas para implementação dos principais conceitos do framework neste problema de otimização.

Movimentos, elementos e mobilidades

Com base nos algoritmos existentes para TR, implementamos dois movimentos básicos que podem ser compostos para gerar os demais. *Shift*, um movimento "horizontal", muda o horário de início de evento para antes ou depois do agendamento atual. *Change*, um movimento "vertical", troca um participante m_i por outro participante m_j de um mesmo grupo g_r, com $m_i, m_j \in g_r$.

Além dos eventos a serem agendados, os próprios participantes foram modelados como elementos do framework. Nas demais aplicações HuGS já implementadas até então, uma única entidade é representada como elemento do problema. Na aplicação HuGS para jobshop, por exemplo, as tarefas são os elementos: os movimentos que operam sobre estes elementos fazem o agendamento das tarefas nas máquinas, não sendo possível realizar movimentos diretamente sobre as máquinas. Como os movimentos shift e change operam sobre dimensões diferentes – eventos e participantes, respectivamente – esta opção de projeto mais complexa foi necessária.

As mobilidades, portanto, também são associadas a estes dois elementos. As mobilidades do evento definem a admissibilidade de movimentos que alterem seu horário. As mobilidades dos participantes nos eventos estabelecem se o participante pode ser trocado por outro de seu grupo naquele evento.

A implementação tira proveito das muitas similaridades do modelo de mobilidades do framework HuGS com os níveis de comprometimento em uma negociação, na qual os eventos e seus participantes podem ser classificados como "confirmados" (mobilidade baixa), "pré-confirmados" (mobilidade média) e "em negociação" (mobilidade alta). Em particular, durante a execução do algoritmo de busca, conferir uma mobilidade mais baixa a um evento de alta prioridade corresponde à estratégia comprometida apresentada na seção 2.5.1.

Scores

Os seguintes critérios foram definidos para comparação dos scores entre duas soluções a e b. Eles representam a função objetivo z do problema TR (3.1), a menos do refinamento

[3]Mais uma vez, cabem agradecimentos ao professor Gunnar Klau, que gentilmente cedeu o código da aplicação para jobshop, originalmente não incluída na distribuição da biblioteca do framework obtida no servidor do MERL – Mitsubishi Electric Research Laboratories.

trazido pelo 4º critério, e são utilizados em ordem, em caso de empate:

1. se a e b são infactíveis e a respeita as restrições de precedência mas b não respeita, então a é melhor do que b; caso contrário, se há pelo menos um slot de tempo de um participante atribuído a dois eventos, e o número de slots infactíveis em a é menor do que em b, então a é melhor do que b;

2. se a é factível e b é infactível, então a é melhor do que b;

3. se a e b são factíveis, mas o número de *participantes* cujas agendas foram afetadas em a é menor do que em b, então a é melhor do que b;

4. se o número de *eventos* reagendados em a é menor do que em b, então a é melhor do que b.

Solução inicial

O algoritmo de busca para TR proposto por Sugihara et al. [77] não agenda, de fato, a $n+1$-ésima tarefa. A partir de $T(n)$, os nós expandidos da árvore de busca apenas rearranjam o agendamento existente, tornando factível o agendamento da nova tarefa sem sobreposições, mas não decidem o horário (τ) e os participantes (ρ) para a tarefa m_{n+1}, deixando esta atribuição a cargo de uma rotina trivial de pós-otimização.

Para um algoritmo controlado visualmente, contudo, esta abordagem não é adequada, pois ela impossibilitaria a identificação das sobreposições entre os eventos existentes e o novo evento a ser agendado. Para explicitar estas infactibilidades, portanto, a solução inicial do algoritmo interativo já deveria consistir de um agendamento $T(n+1) = (P, M_{n+1}, <', t, p, w, \tau', \rho')$.

Definimos, então, um agendamento trivial como:

- $\tau'(m_{n+1}) = t_1(m_{n+1})$, isto é, a primeira instância de tempo em $t(n+1)$ na qual m_{n+1} pode ser agendado; e

- $\rho'(n+1) = \cup_{g_r \in p(m_{n+1})} g_{1r}$, isto é, as primeiras pessoas de cada grupo g_r de participantes em $p(m_{n+1})$.

O processo de otimização tem início a partir do nó raiz, que representa a solução com agendamento trivial $T(n+1)$ no grafo de busca. A cada iteração, são gerados todos os movimentos shift e change admitidos pela instância do problema e pelas mobilidades correntes.

A figura 5.2 mostra a solução inicial para a instância de TR apresentada no exemplo 2 da seção 3.5.1. Note o agendamento trivial do novo evento m_6 (‘*** MEETING 6 ***’). Os horários do evento m_5 (‘Meeting 5’) marcados em vermelho denotam sua baixa mobilidade,

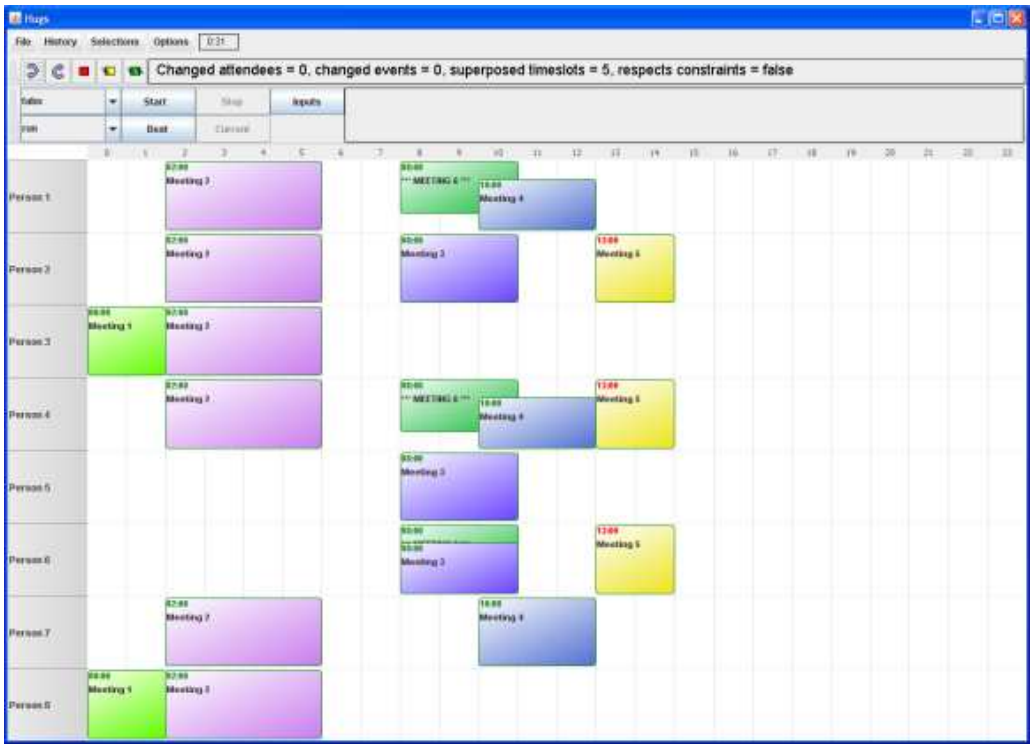


Figura 5.2: Execução do algoritmo com instância $T(5)$ da figura 3.22

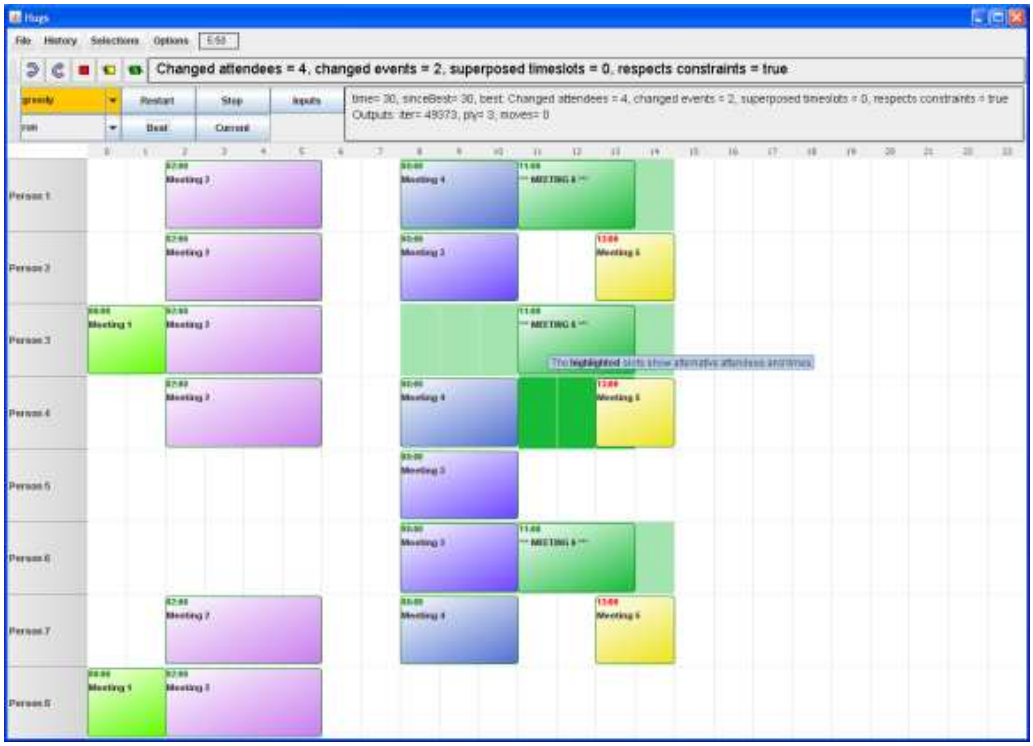


Figura 5.3: Solução ótima $T(6)$ da figura 3.23

já que $F = \{m_5\}$ é o conjunto de eventos cujos horários de início estão fixados. As bordas verdes dos eventos representados na agenda de cada participante denotam a mobilidade do participante no evento.

A figura 5.3 mostra a solução ótima $T(6)$ obtida após o processo de busca.

Visualização

O componente MiG Calendar[4], que oferece uma biblioteca Java para a criação de componentes visuais de calendários baseados em metáforas de agendas reais, foi utilizado para implementar a visualização do algoritmo interativo. Com isso, o protótipo herdou recursos avançados – como drag-and-drop para realização dos movimentos shift e change – que tornam sua interface semelhante à das aplicações desktop típicas.

Para facilitar comparações entre instâncias de TR, optamos por implementar uma interface semelhante à representação utilizada nos artigos de Sugihara et al. [77] e Sugumaran et al. [78]. Os participantes são organizados verticalmente e o eixo horizontal corresponde à dimensão tempo.

A camada de visualização conta com facilidades adicionais oferecidas pela biblioteca. Na figura 5.3, por exemplo, os horários disponíveis para movimentos shift e os participantes candidatos a movimentos change são destacados quando o mouse é apontado sobre um evento na agenda de um participante.

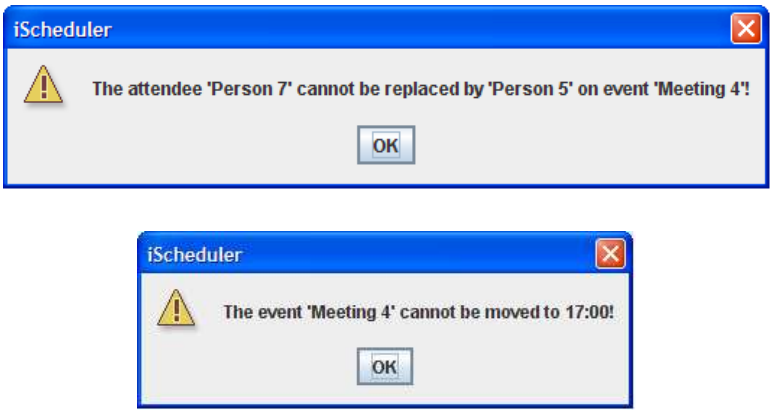


Figura 5.4: Mensagens de aviso para movimentos shift e change inválidos

O sofisticado mecanismo de captura de eventos do componente MiG Calendar também permitiu que algumas consistências relativas às ações dos usuários fossem implementadas. A figura 5.4 ilustra as mensagens de aviso emitidas quando movimentos manuais inválidos são executados.

[4] `http://www.migcalendar.com/`.

Extensão para versão generalizada de TR

Na verdade, os conceitos do framework HuGS foram implementados para uma **versão generalizada** de TR, que consiste em:

- um agendamento $T(n) = (P,\ M_n,\ <,\ t,\ p,\ w,\ \tau,\ \rho)$,
- um *conjunto* $M_k = \{n+1,\ ...,\ n+k\}$ de k eventos adicionais a serem agendados,
- $t(m_k)$, $p(m_k)$ e $w(m_k)$,
- uma ordem parcial $<'$ em $M_{n+k} = M_n \cup M_k$, e
- um conjunto $F(\subseteq M_n)$ de eventos cujos horários de início estão fixados,

cujo objetivo é encontrar um agendamento $T(n+k) = (P,\ M_{n+k},\ <',\ t,\ p,\ w,\ \tau',\ \rho')$ que minimiza a mesma função objetivo z definida para o problema TR original (3.1).

A figura 5.5 ilustra a solução ótima de uma instância da versão generalizada de TR com $k = 2$, para a qual os novos eventos são m_6, da instância já examinada, e m_7, com $t(m_7) = 3$, $p(m_7) = \{\{3,\ 8\},\ \{5\},\ \{6\}\}$ e $w(m_7) = \{0\}$.

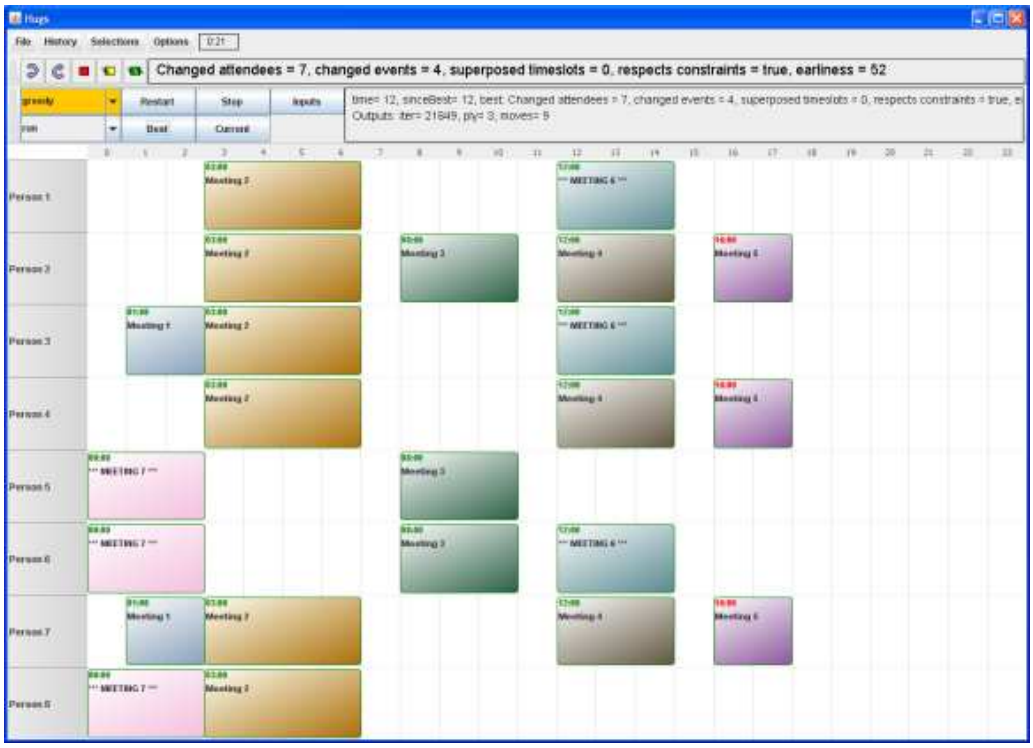


Figura 5.5: Solução ótima para uma instância da versão generalizada de TR

Note que, com $n = 0$, a versão generalizada de TR consiste em, simplesmente, agendar k eventos. Neste caso, qualquer solução factível tem um valor ótimo de z. A arquitetura

do framework HuGS permite, com a implementação do score, comparar duas soluções utilizando quaisquer critérios, sendo possível utilizá-la para representar qualquer função objetivo que se queira. Com isto, obtém-se um algoritmo de (re-)agendamento genérico.

Extensão para mecanismo de preferências

O conceito de score comporta, inclusive, a implementação de um critério de comparação baseado nas preferências de agendamento dos participantes. Alimentado por uma base de conhecimento apropriada, o processo de busca poderia considerá-las.

Uma pequena modificação foi feita nos critérios de comparação de scores para demonstrar a possibilidade de se adotar um mecanismo de inteligência artificial, como o descrito na seção 3.6.1, em conjunto com um algoritmo de otimização interativa para agendamento. Adicionamos o seguinte critério de comparação para duas soluções a e b:

5. Se os eventos em a começam mais cedo (tarde) do que em b, então a é melhor do que b.

O cálculo realizado pelo algoritmo é bastante simples, mas suficiente para este propósito: os horários de início dos eventos nas soluções a e b são somados. Dependendo da preferência de horários (agendamentos mais cedo ou mais tarde), a e b são ordenados de acordo com o valor desta soma.

As figuras 5.6 e 5.7 mostram soluções alternativas para $T(6)$, de mesmo custo de rearranjo, mas que satisfazem a diferentes preferências de horários. No primeiro caso, quanto mais cedo agendados os eventos, melhor a solução; no segundo caso, o oposto.

5.3 Resultados

Através do desenvolvimento de uma biblioteca com suporte aos componentes propostos como extensão para o padrão iCalendar, foi possível perceber as dificuldades de se chegar a uma implementação interoperável para validação de objetos iCalendar. Isto nos ajudou a compreender melhor a tendência à simplificação por parte dos desenvolvedores de protocolos, que muitas vezes têm uma posição de não implementar uma funcionalidade se houver muitas dificuldades em garantir sua compatibilidade.

A biblioteca do framework HuGS permitiu uma implementação relativamente rápida e sofisticada para TR. Não foram realizados testes extensivos com famílias de instâncias para medir o seu desempenho[5], embora as soluções ótimas para as instâncias apresentadas sejam obtidas quase que instantaneamente. Também são necessários, ainda, testes para verificar o grau de usabilidade da interface e as modificações que seriam necessárias para

[5]O apêndice B detalha características da arquitetura da biblioteca HuGS que impõem dificuldades à medição de desempenho dos algoritmos.

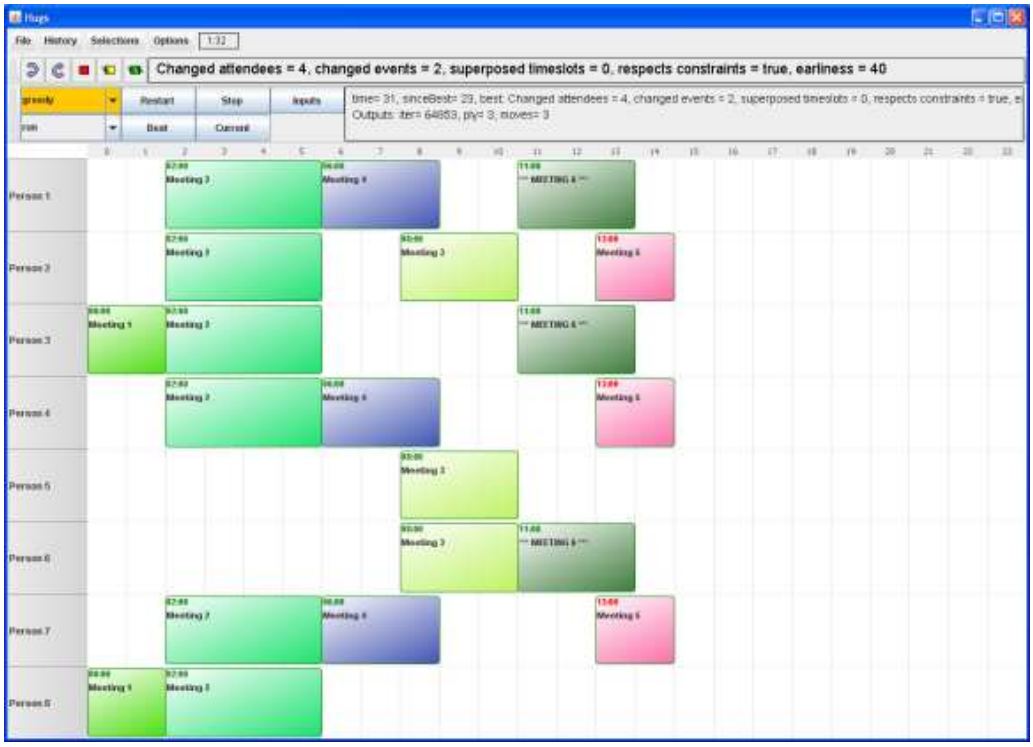


Figura 5.6: Agendamento privilegiando horários mais cedo

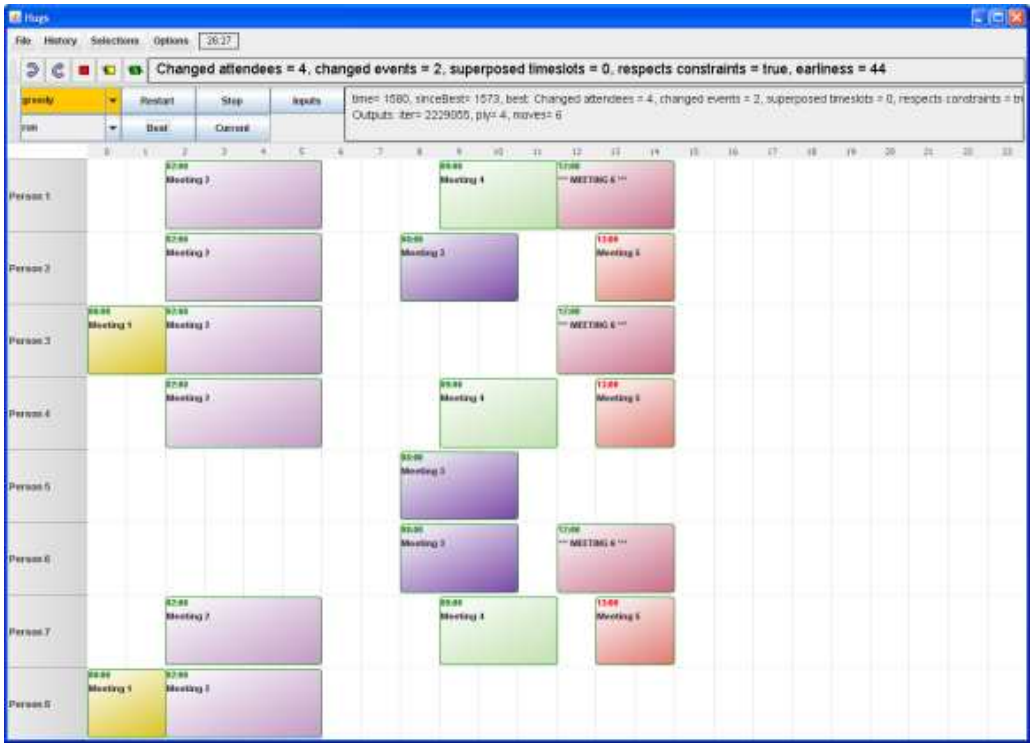


Figura 5.7: Agendamento privilegiando horários mais tarde

adaptá-la a um ambiente de uso real, bem como o impacto de um formato de visualização distinto daquele adotado pelos sistemas de agendamento mais populares em utilização. Neste sentido, o protótipo já obtido é um excelente ponto de partida.

Embora estas implementações não estejam integradas, acreditamos que elas cumprem seu papel ao oferecer uma prova de conceito com relação aos dois pontos fundamentais nesta proposta de visão interdisciplinar de agendamento em grupo: (1) é possível acrescentar funcionalidades aos protocolos padrão através de extensões bem especificadas que não comprometam a interoperabilidade com aplicações já existentes; e (2) a integração de técnicas de otimização (e inteligência artificial) a aplicações de agendamento em grupo pode ser viabilizada através de uma abordagem híbrida homem-máquina.

Capítulo 6

Considerações finais

Com o prenunciado sucesso do protocolo CalDAV em estabelecer-se como o padrão para compartilhamento de calendários e agendamento, devido ao grande suporte dos maiores fabricantes de software, espera-se que os problemas de interoperabilidade existentes sejam mitigados em breve. Assim, os usuários tenderão a ganhar mais liberdade para escolher sua aplicação de agendamento, mesmo em ambiente corporativo.

A partir do momento em que o agendamento em grupo entre diversos sistemas for realidade, haverá demandas por executá-lo com mais qualidade e eficiência. O próximo passo a ser tomado pelos fabricantes de software será dado em direção a aplicações mais poderosas e inteligentes, capazes de automatizar tarefas repetitivas e reduzir a carga de trabalho cognitiva dos usuários, e tirando proveito dos ganhos de produtividade latentes no uso de dispositivos como celulares e palmtops. Enquanto algumas funcionalidades adicionais não dependerão de mudanças semânticas na comunicação, outras poderão demandar avanços nos protocolos para que possam funcionar efetivamente em um ambiente de trabalho coletivo.

O processo de desenvolver protocolos não tem o mesmo dinamismo do mercado de software, que muda muito rapidamente. Ao mesmo tempo em que tende a gerar soluções robustas, por ser influenciada por diversos interessados, a especificação de padrões abertos através de um processo como o da IETF está sujeita a discordâncias conceituais e tecnológicas que consomem tempo até serem decantadas em decisões de projeto consensuais. Concordamos que estes podem ser fatores de desestímulo, mas também identificamos na falta de aderência aos padrões uma das principais razões pelas quais diversas soluções desenvolvidas até o momento, a despeito de sua inegável relevância, não foram bem-sucedidas. Formas proprietárias e incompatíveis de compartilhamento de informações não devem mais ser adotadas. Sob os auspícios de uma organização como o consórcio CalConnect, processos de padronização prementes podem ser conduzidos mais rapidamente em benefício de todos.

Tanenbaum [82] argumenta que o momento ideal para o desenvolvimento de protocolos

está entre os "dois elefantes" da figura 6.1; isto é, as pesquisas já realizadas garantem que os padrões não serão estabelecidos precocemente, com uma compreensão incompleta dos problemas a serem resolvidos, e nem após grandes investimentos da indústria de software, de forma que caminhos muito diferentes dos padrões propostos tenham sido tomados e as tentativas de estabelecer padrões sejam inócuas.

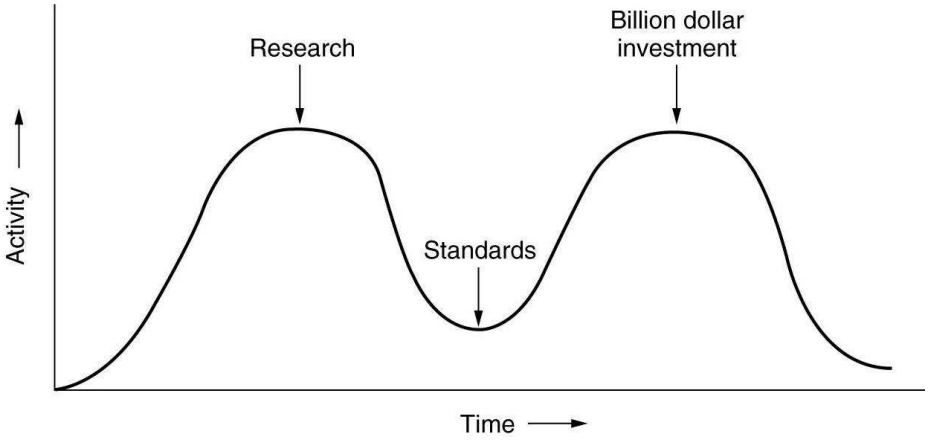


Figura 6.1: Os "dois elefantes" e o desenvolvimento de padrões [82]

Acreditamos que este é o momento entre os "dois elefantes" para as ferramentas de agendamento de terceira geração: ao mesmo tempo em que um volume considerável de pesquisas e discussão sobre agendamento em grupo foi realizado, e há uma experiência consolidada com o desenvolvimento dos sistemas de segunda geração que permite compreender melhor as necessidades dos usuários, o desenvolvimento maciço do que serão as ferramentas mais flexíveis e inteligentes ainda não começou. Portanto, este é o momento de estabelecer os protocolos que viabilizarão a implementação das funcionalidades que serão demandadas num futuro breve.

Uma abordagem integrada implica em considerar as relações entre esses protocolos de comunicação e os diversos mecanismos que estarão no núcleo da próxima geração de sistemas de agendamento. As experiências passadas demonstram que o foco em uma única área de pesquisa leva a futuros problemas na adoção das soluções propostas na prática, devido às influências mútuas de diferentes áreas de conhecimento entre si. Otimização, inteligência artificial e técnicas de interação não podem, isoladamente, prover uma definitiva solução para agendamento em grupo, mas certamente oferecem a perspectiva de avanços consideráveis se aplicadas conjuntamente.

Advogamos por um relevante direcionamento: os trabalhos futuros de pesquisa e desenvolvimento em agendamento em grupo não devem considerar estas disciplinas separadamente, mas avaliar de forma cuidadosa as questões que surgem na tentativa de conciliá-las. Deve-se empregar um esforço considerável não somente no desenvolvimento de inovações, mas também no tratamento das implicações de sua adoção em um ambiente heterogêneo.

6.1 Contribuições

Consideramos como relevante produto deste trabalho a revisão bibliográfica sobre as aplicações para compartilhamento de calendários e agendamento. Com base nela, foi possível identificar a necessidade de novos rumos para o desenvolvimento das futuras aplicações e estabelecer uma abordagem interdisciplinar como solução para enfrentar os desafios do complexo problema que é agendamento em grupo.

Ao desenvolver extensões para os protocolos iCalendar, e participar das listas de discussão de protocolos e de algumas das discussões semanais do grupo Calsify, em que são discutidas as versões "bis" das RFCs que especificam os protocolos em vigor, foi possível compreender a postura conservadora dos implementadores, já que opções de projeto mais simplificadas favorecem implementações com menos problemas de interoperabilidade. Este ponto de vista é reforçado pelas dezenas de *issues* identificados nas RFCs atuais e cujo tratamento está sendo finalizado, mas não acreditemos, contudo, que tais dificuldades justifiquem a não-implementação de novas funcionalidades.

De alguma forma, tomamos parte de um momento interessante na evolução dos protocolos, quando a diretriz de simplificação em favor da interoperabilidade colocou em questão algumas funcionalidades que consideramos fundamentais nos protocolos. Quando se cogitou, por exemplo, abolir o método `COUNTER`, devido ao baixo número de aplicações que o implementam, nos juntamos a outros desenvolvedores também contrários a esta proposta. Tivemos oportunidade, também, de argumentar a favor da manutenção do mecanismo de seqüenciamento do protocolo iTIP, por considerá-lo indispensável para protocolos de transporte *store-and-forward* como iMIP.

Durante a implementação dos novos componentes iCalendar que propomos, também foi possível perceber na prática as dificuldades de garantir sua validação correta. Mesmo com base em uma hierarquia de classes tão bem estruturada como a da biblioteca iCal4j, para a qual sugerimos a correção de um bug a este respeito, e algumas modificações que pudessem torná-la mais apropriada para a implementação de extensões[1].

Com uma implementação interativa para Timetable Rearrangement, demonstramos que algoritmos de otimização, possivelmente integrados a mecanismos de inteligência artificial, podem ser utilizados em aplicações de agendamento em grupo. O próprio êxito da aplicação do framework HuGS em um problema pouco estudado reitera a relevância da adoção de arquiteturas de iniciativa mista, inclusive para outros problemas de otimização.

6.2 Trabalhos futuros

Embora os resultados da utilização de um algoritmo de otimização associado a uma interface de manipulação direta indiquem a viabilidade de adotá-los em aplicações de agen-

[1] `http://sourceforge.net/forum/forum.php?thread_id=1803020&forum_id=368290`.

damento em grupo, estudos aprofundados com relação ao impacto de sua incorporação são necessários. Além disso, a eficiência de um agente de iniciativa mista em um fluxo de agendamento não dependente de emails também está por ser analisada.

Diante de certo ceticismo dos desenvolvedores dos protocolos da família iCalendar com relação à proposta de adoção de extensões mais flexíveis (percebido em parte pelas opiniões a respeito da dificuldade de construir aplicações interoperáveis com elas, e em parte pelo próprio baixo volume de comentários recebidos nas listas de discussão),

Acreditamos que um caminho de grande impacto para impulsionar a terceira geração de sistemas de agendamento – e de seus protocolos – seria uma implementação completa e de código aberto do iScheduler. Esta implementação contribuiria, inclusive, para o refinamento das propostas de extensão dos padrões atuais que especificamos, eventualmente possibilitando sua evolução para uma RFC.

Para reduzir o considerável esforço de desenvolvimento que esta empreitada exigiria, a implementação poderia ser baseada em uma extensão do projeto de código aberto Columba[2], um cliente de email independente de plataforma desenvolvido em Java, para o qual uma aplicação de calendários está sendo desenvolvida. Esta aplicação utiliza as mesmas bibliotecas MiG Calendar e iCal4j às quais recorremos.

Um dos servidores CalDAV de código aberto existentes também poderia ser modificado para suportar as extensões para iCalendar e iTIP. Cosmo[3] e Bedework[4] poderiam ser considerados por serem desenvolvidos em Java, o que possibilitaria o reaproveitamento do código que desenvolvemos para a biblioteca de entrada e saída de dados. CalDAV4j[5], uma biblioteca cliente para a execução de operações com um servidor CalDAV, poderia ser adaptada sem muitas dificuldades para suportar os novos componentes propostos, já que também utiliza a biblioteca iCal4j.

Finalmente, algumas limitações da biblioteca do framework HuGS, detalhadas no apêndice B, indicam a conveniência de reestruturar sua hierarquia de classes, possivelmente incorporando conceitos do framework User Hints, a fim de viabilizar sua utilização integrada na aplicação[6]. Tal reestruturação seria benéfica não só para um projeto de implementação do iScheduler, mas para outros problemas de otimização combinatória que necessitem de uma biblioteca que implemente um framework de otimização interativa.

Esta abundância de possíveis desdobramentos demonstra a grande diversidade de desafios a serem enfrentados. Embora o presente trabalho seja apenas o estágio inicial de uma nova abordagem, esperamos que ele seja capaz de indicar que o êxito de iniciativas futuras em agendamento em grupo depende da avaliação das questões que envolvem o tema sob uma nova perspectiva.

[2] `http://columbamail.org/`.
[3] `http://chandlerproject.org/Projects/CosmoHome`.
[4] `http://www.bedework.org/`.
[5] `http://code.google.com/p/caldav4j/`.
[6] Além disso, a licença de uso do HuGS não permite seu uso em escala, mas somente para pesquisa.

Apêndice A

Especificação das extensões para iCalendar e iTIP

Este apêndice traz a especificação das extensões dos protocolos iCalendar e iTIP submetida para o diretório de drafts da IETF. Ela também está disponível no endereço `http://www.ietf.org/internet-drafts/draft-silva-events-01.txt`.

Em relação à versão -00 do draft, que havia sido submetida em 29/05/2007, esta versão -01, de 20/12/2007, traz a definição completa dos métodos iTIP para os novos componentes, que haviam sido somente esboçados na versão anterior, e melhorias com respeito à representação de tarefas líquidas[1].

A especificação tem todas as características exigidas de uma RFC [64], como o formato somente texto e a estrutura determinada pela IETF. Note-se que o foco de uma especificação como esta consiste menos em uma descrição argumentativa sobre as vantagens de sua adoção e mais em um documento formal de referência para implementação de sistemas interoperáveis. Como tal, ela faz uso de notações específicas, a fim de mitigar ambigüidades e garantir sua consistência.

Embora para a total compreensão de seu conteúdo seja necessário o conhecimento dos drafts das versões "bis" das RFCs 2445 (iCalendar) [23] e 2446 (iTIP) [17], da RFC 4791 (CalDAV) [20], e do draft para as extensões de agendamento em CalDAV [19], os elementos básicos para seu entendimento presentes nestes padrões encontram-se na seção 3.3[2].

Esta especificação de extensões segue os guidelines para Internet-Drafts [31], e já atende ao checklist necessário aos drafts candidatos para publicação como RFC [84].

---

[1] As diferenças entre as duas versões podem ser verificadas no endereço `http://tools.ietf.org/html/draft-silva-events-01`.

[2] A especificação herda da RFC 2445 a notação Augmented Backus-Naur Form (ABNF) – RFC 4234 [16] – utilizada na descrição sintática dos elementos do padrão iCalendar, e da RFC 2446 as *restriction tables* que descrevem as restrições específicas impostas a componentes, propriedades e parâmetros por cada método iTIP. Contudo, os exemplos já apresentados tornam dispensáveis maiores detalhes a respeito.

Network Working Group F. Silva
Internet-Draft R. Drummond
Intended status: Experimental Unicamp
Expires: June 22, 2008 December 20, 2007

Imprecise and alternative events for iCalendar and iTIP
draft-silva-events-01

Status of this Memo

Copyright Notice

Abstract

This document defines extensions to iCalendar and iTIP that allow the expression of imprecise and multiple alternative events in calendar objects and scheduling messages. Imprecise events can be used to imprecisely describe an event by specifying various ranked periods of time in which it could be scheduled. Alternative events can be used to describe an event in terms of multiple ranked event descriptions, imprecise or not, that could alternatively be scheduled.

Editorial note (to be removed by RFC Editor prior to publication)

Discussion of this specification is taking place on the mailing list http://lists.osafoundation.org/mailman/listinfo/ietf-calsify.

Table of Contents

1. Introduction
2. Conventions Used in This Document
3. iCalendar Extensions
  3.1. Imprecise Event Component
  3.2. Alternative Events Component
  3.3. Minimum Granularity Component Property
  3.4. Maximum Granularity Component Property
  3.5. Rank Component Property
  3.6. Rank Property Parameter
4. iTIP Extensions
  4.1. Methods for VIMPRECISEEEVENT Calendar Components
    4.1.1. PUBLISH
    4.1.2. REQUEST
    4.1.3. REPLY
    4.1.4. CANCEL
    4.1.5. REFRESH
    4.1.6. COUNTER
    4.1.7. DECLINECOUNTER
  4.2. Methods for VALTERNATIVEEVENTS Calendar Components
    4.2.1. PUBLISH
    4.2.2. REQUEST
    4.2.3. REPLY
    4.2.4. CANCEL
    4.2.5. REFRESH
    4.2.6. COUNTER
    4.2.7. DECLINECOUNTER
5. Examples
  5.1. Published Imprecise Event
  5.2. An Imprecise Event Request
  5.3. Reply to an Imprecise Event Request
  5.4. Counter an Imprecise Event Request
  5.5. Update an Event Description
6. IANA Considerations
7. Security Considerations
8. Acknowledgements
9. References
  9.1. Normative References
  9.2. Informative References

Appendix A. Design Considerations
A.1. Avoiding VALTERNATIVEEVENTS
A.2. Avoiding VIMPRECISEEVENT
A.3. Compatibility
Appendix B. Change History (to be removed by RFC Editor prior to publication)
Authors' Addresses
Intellectual Property and Copyright Statements

1. Introduction

Calendar users may want to express an event not by using a unique exact description, but in terms of several possible alternative event descriptions. A possible event could, in turn, be described imprecisely, i.e., not with fixed start and end times, but in terms of one or more periods of time in which it could occur. Additionally, it should be possible to rank each of these alternative events and periods of time, so as to represent the suitability or likelihood of scheduling them in fact.

These features would make possible the description of an event like the following: a 1 hour long business meeting taking place at office A on Monday between 8 am and 11 am - preferred -, or at office B on Thursday 2 pm or Friday after 4:30 pm. Such flexible representation of an event allow users to communicate uncertainty and preferences that naturally occur [SL02].

This is especially important during scheduling negotiation, when the "Attendees" have to agree about an event description. Multiple alternative event descriptions and periods of time can help a group of "Attendees" to faster converge into a description that is acceptable for all [SD98]. In most situations, that means to reach consensus about an event time that satisfies a common feasible free time interval, but other event properties can be under negotiation as well (e.g., duration, location, summary). In the worst case, if only the nth event proposal sent by the "Organizer" is acceptable by all "Attendees", offering one event description per proposal leads to n negotiation 'rounds'. Conversely, in the best case, if the "Organizer" had proposed n alternative event descriptions, only one negotiation round would have been enough.

However, iCalendar [I-D.ietf-calsify-rfc2445bis] does not provide a way to describe an event imprecisely or to represent multiple event alternatives. This cannot be done by using several iTIP [I-D.ietf-calsify-2446bis] messages either.

To accomplish these, this memo defines new types of iCalendar calendar components and specifies iTIP methods for their use in scheduling messages.

2. Conventions Used in This Document

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].

"VEVENT" components defined by [I-D.ietf-calsify-rfc2445bis] are referred to as "regular events". The term "event" is used as a general reference to regular events, imprecise events, and alternative events.

3. iCalendar Extensions

This specification adds the new "VIMPRECISEEVENT" and "VALTERNATIVEEVENTS" calendar components to iCalendar. The "VIMPRECISEEVENT" component describes an imprecise event in terms of periods of time in which it could be scheduled. These periods are expressed by "VFREEBUSY" and "VAVAILABILITY" [I-D.daboo-calendar-availability] components. The "VALTERNATIVEEVENTS" component describes alternative events by grouping multiple "VEVENT" and "VIMPRECISEEVENT" event descriptions.

Whereas the "VIMPRECISEEVENT" and "VALTERNATIVEEVENTS" components describe events, they are not intended to represent regular events that correspond to a scheduled amount of time, what is done by the "VEVENT" component. Imprecise and alternative events components do not take up time on a calendar, since they just represent a schedulable amount of time whose description definition is in course. As such, these components MUST NOT be considered to generate results for an iTIP "VFREEBUSY" REQUEST, what includes CalDAV calendar-access [RFC4791] CALDAV:free-busy-query REPORT and CalDAV calendar-schedule [I-D.desruisseaux-caldav-sched] free-busy lookup.

This specification also adds the new "RANK" property to the "VEVENT" and "VIMPRECISEEVENT" components, and to the "AVAILABLE" sub-components of the "VAVAILABILITY" component. A new "RANK" property parameter is also defined for the "FREEBUSY" parameter of the "VFREEBUSY" component. These extensions allow alternative events and periods of time used to describe imprecise events to be ranked.

Moreover, the new "MIN-GRANULARITY" and "MAX-GRANULARITY" properties

```
   are introduced for the "VIMPRECISEEVENT" component.  If an imprecise
   event represents some kind of "liquid task" that can be scheduled in
   more than one regular event, these properties indicate, respectively,
   the minimum and maximum duration that each of such events should
   have.

3.1.  Imprecise Event Component

   Component Name:  VIMPRECISEEVENT

   Purpose:  Provide an imprecise event description.

   Format Definition:  A "VIMPRECISEEVENT" calendar component is defined
      by the following notation:

        impreciseeventc     = "BEGIN" ":" "VIMPRECISEEVENT" CRLF
                              impreciseeventprop *availabilityc
                              *freebusyc
                              "END" ":" "VIMPRECISEEVENT" CRLF

        impreciseeventprop  = *(

                              ; the following are REQUIRED,
                              ; but MUST NOT occur more than once

                              dtstamp / uid /

                              ; the following are OPTIONAL,
                              ; but MUST NOT occur more than once

                              class / created / description / due /
                              duration / geo / last-mod / location /
                              min-granularity / max-granularity /
                              organizer / priority / rank / rrule /
                              seq / status / summary / transp / url /

                              ; 'percent' is OPTIONAL,
                              ; but MUST NOT occur more than once and,
                              ; if it occurs, 'duration' MUST occur

                              percent /

                              ; the following are OPTIONAL,
                              ; and MAY occur more than once

                              attach / attendee / categories / comment /
                              contact / rstatus / related / resources /
```

```
                        x-prop / iana-prop

                        )
```

Description: A "VIMPRECISEEVENT" calendar component is a grouping of component properties, and possibly including "VFREEBUSY" and "VAVAILABILITY" components, that represents a schedulable amount of time, imprecisely described, for which exact start and end times are not specified. As opposed to a regular event represented by a "VEVENT" component, a "VIMPRECISEEVENT" does not represent a scheduled amount of time on a calendar, but an imprecise event that, once having its temporal properties - start time and end time (or duration) - defined, can be represented by a "VEVENT" component.

Included "VFREEBUSY" and "VAVAILABILITY" components represent available periods of time for effectively scheduling the event. The "VFREEBUSY" components MUST follow the same semantics used to publish busy time specified on [I-D.ietf-calsify-rfc2445bis]. Typically, the "FBTYPE" parameter of "FREEBUSY" properties present on these "VFREEBUSY" components will have the "FREE" value, but "BUSY-TENTATIVE" and "BUSY" values MAY also be used so as to denote that events already scheduled on the interval represented by the "FREEBUSY" property could be rescheduled in favor of the imprecise event in question. The "BUSY-UNAVAILABLE" value SHOULD NOT be used for this purpose.

The resulting available time is the union of the periods of time represented by all "VFREEBUSY" and "VAVAILABILITY" components. Periods of time specified by the "VFREEBUSY" components MUST NOT overlap. However, a period specified by a "VFREEBUSY" component MAY overlap those represented by "VAVAILABILITY" components. In this case, the interval specified via "VFREEBUSY" overrides those defined by any "VAVAILABILITY" components.

Besides the applicable properties from the "VEVENT" component, the "VIMPRECISEEVENT" component also allows for the "DUE" and "PERCENT" properties present on the iCalendar "VTODO" component. As in a to-do, the "DUE" property defines the date and time that the event is expected to be finished. The "PERCENT" property specifies, in the case of a liquid task, how much of it has already been scheduled or accomplished into regular events, with respect to the "DURATION" property. From that, the remaining duration that still must be scheduled can be inferred.

In the context of an imprecise event, a "RRULE" is not intended to specify a concrete recurrence set of events, but to express a

```
   frequency of occurrence that the actual scheduled event may have.
   Within a "VIMPRECISEEVENT" component the UNTIL rule part of a
   recurrence rule MUST NOT be specified.

Example:  The following is an example of a "VIMPRECISEEVENT" calendar
   component that describes a 2 hour long department meeting taking
   place whenever between 8:00 AM to 12:00 on April 1st, 2008:

     BEGIN:VIMPRECISEEVENT
     UID:20071005T133225Z-00001@example.com
     DTSTAMP:20071005T133225Z
     SUMMARY:Department meeting
     DURATION:PT2H
     BEGIN:VFREEBUSY
     UID:20071005T133225Z-00001A@example.com
     DTSTAMP:20071005T133225Z
     FREEBUSY;FBTYPE=FREE:20080401T080000Z/20080401T120000Z
     END:VFREEBUSY
     END:VIMPRECISEEVENT

   The following is an example of a "VIMPRECISEEVENT" calendar
   component that describes a medical check-up occurring within any
   of the periods of time specified by the "VFREEBUSY" components and
   that is intended to have a monthly recurrence:

     BEGIN:VIMPRECISEEVENT
     UID:20071005T133225Z-00001@example.com
     DTSTAMP:20071005T133225Z
     SUMMARY:Medical check-up
     RRULE:FREQ=MONTHLY;BYDAY=MO,TU,WE,TH,FR
     BEGIN:VFREEBUSY
     UID:20071005T133225Z-00001-A@example.com
     DTSTAMP:20071005T133225Z
     FREEBUSY;FBTYPE=FREE;RANK=100:20080328T150000Z/PT3H
     FREEBUSY;FBTYPE=FREE;RANK=100:20080329T163000Z/PT1H30M
     FREEBUSY;FBTYPE=FREE;RANK=80:20080330T140000Z/PT6H
     END:VFREEBUSY
     BEGIN:VFREEBUSY
     UID:20071005T133225Z-00001-B@example.com
     DTSTAMP:20071005T133225Z
     COMMENT:If necessary, I can try to rearrange my agenda
     FREEBUSY;FBTYPE=BUSY-TENTATIVE:20080401T160000Z/PT1H
     END:VFREEBUSY
     END:VIMPRECISEEVENT

   The following is an example of a "VIMPRECISEEVENT" calendar
   component that describes a dental treatment with total duration of
```

10 hours, for which 20% have been completed. The remaining 8 hours can be scheduled on appointments of at least 1 hour and no more than 2 hours. The "VAVAILABILITY" component specifies a 4:00 PM to 6:00 PM recurring availability on Tuesdays and Thursdays, from March 15th to April 15th. The "VFREEBUSY" component also overrides the "VAVAILABILITY" component between 4:00 PM and 5:00 PM on April 1st and specifies an additional free interval from 6:00 PM to 7:00 PM on April 3rd.

```
BEGIN:VIMPRECISEEVENT
UID:20071005T133225Z-00001@example.com
DTSTAMP:20071005T133225Z
SUMMARY:Dentist
DURATION:PT10H
PERCENT-COMPLETE:20
MIN-GRANULARITY:PT1H
MAX-GRANULARITY:PT2H
BEGIN:VAVAILABILITY
UID:20071005T133225Z-00001-A@example.com
DTSTAMP:20071005T133225Z
DTSTART:20080315T160000Z
BEGIN:AVAILABLE
UID:20071005T133225Z-00001-A1@example.com
RANK:95
DTSTART:20080315T160000Z
DTEND:20080415T180000Z
RRULE:FREQ=WEEKLY;BYDAY=TU,TH
END:AVAILABLE
END:VAVAILABILITY
BEGIN:VFREEBUSY
UID:20071005T133225Z-00001-B@example.com
DTSTAMP:20071005T133225Z
FREEBUSY;FBTYPE=BUSY;RANK=33:20080401T160000Z/PT1H
FREEBUSY;FBTYPE=FREE;RANK=60:20080403T180000Z/PT1H
END:VFREEBUSY
END:VIMPRECISEEVENT
```

3.2. Alternative Events Component

Component Name: VALTERNATIVEEVENTS

Purpose: Provide a grouping of event components ("VEVENT" and "VIMPRECISEEVENT") that represent alternative event descriptions.

Format Definition: A "VALTERNATIVEEVENTS" calendar component is defined by the following notation:

```
     alternativeeventsc     = "BEGIN" ":" "VALTERNATIVEEVENTS" CRLF
                              alternativeeventsprop *eventc
                              *impreciseeventc
                              "END" ":" "VALTERNATIVEEVENTS" CRLF

     alternativeeventsprop  = *(

                              ; the following are REQUIRED,
                              ; but MUST NOT occur more than once

                              dtstamp / uid /

                              ; the following are OPTIONAL,
                              ; but MUST NOT occur more than once

                              class / created / description / geo /
                              last-mod / location / organizer /
                              priority / seq / status / summary /
                              transp / url /

                              ; the following are OPTIONAL,
                              ; and MAY occur more than once

                              attach / attendee / categories / comment /
                              contact / rstatus / related / resources /
                              x-prop / iana-prop

                              )
```

Description: A "VALTERNATIVEEVENTS" calendar component is a grouping of component properties, including at least two event components, that represents a schedulable amount of time, described in terms of alternative event descriptions, for which exact property values are not specified. As opposed to a regular event represented by a "VEVENT" component, a "VALTERNATIVEEVENTS" does not represent a scheduled amount of time on a calendar, but an event with multiple possible alternatives that, once having its properties defined, can be represented by a "VEVENT" component.

If specified on a "VALTERNATIVEEVENTS" component, OPTIONAL properties SHOULD also have their values interpreted as the default values for the properties of each grouped event component. In this case, if also present on a grouped event component, an OPTIONAL property overrides its default value.

Example: The following is an example of a "VALTERNATIVEEVENTS" calendar component used to represent an event that can take one of

```
the descriptions represented by the "VEVENT" and "VIMPRECISEEVENT"
components.  Note that the "COMMENT" property default value
specified by the alternative events component is overridden within
the regular event component.

BEGIN:VALTERNATIVEEVENTS
UID:20071005T133225Z-00001@example.com
DTSTAMP:20071005T133225Z
SUMMARY:Quick interop test
COMMENT:Trying the options that seem most likely to be accepted...
BEGIN:VEVENT
UID:20071005T133225Z-00001-A@example.com
DTSTAMP:20071005T133225Z
DTSTART:20080406T090000Z
DTEND:20080405T120000Z
RANK:100
COMMENT:It would be perfect if we could meet at that same time!
END:VEVENT
BEGIN:VIMPRECISEEVENT
UID:20071005T133225Z-00001-B@example.com
DTSTAMP:20071005T133225Z
RANK:70
BEGIN:VAVAILABILITY
UID:20071005T133225Z-00001-B-A@example.com
DTSTART:20080315T160000Z
COMMENT:I myself could attend at any of these times.
BEGIN:AVAILABLE
UID:20071005T133225Z-00001-B-A1@example.com
RANK:80
DTSTART:20080315T140000Z
DTEND:20080415T180000Z
RRULE:FREQ=WEEKLY;BYDAY=TU,TH
END:AVAILABLE
END:VAVAILABILITY
BEGIN:VFREEBUSY
UID:20071005T133225Z-00001-B-B@example.com
DTSTAMP:20071005T133225Z
FREEBUSY;FBTYPE=BUSY;RANK=33:20080401T160000Z/PT1H
FREEBUSY;FBTYPE=FREE;RANK=60:20080403T180000Z/PT1H
END:VFREEBUSY
END:VIMPRECISEEVENT
END:VALTERNATIVEEVENTS
```

3.3. Minimum Granularity Component Property

```
Property Name:  MIN-GRANULARITY
```

```
   Purpose:  This property is used to convey the minimum granularity of
      an imprecise event.

   Value Type:  DURATION

   Property Parameters:  IANA and non-standard property parameters can
      be specified on this property.

   Conformance:  This property can be specified once in a
      "VIMPRECISEEVENT".

   Description:  This property defines the minimum duration that an
      event to be scheduled in order to accomplish a liquid task
      described by an imprecise event must have.

   Format Definition:  This property is defined by the following
      notation:

           min-granularity  = "MIN-GRANULARITY" min-granulparam ":"
                              dur-value CRLF

           min-granulparam  = *(";" other-param)

   Example:  The following is an example of this property to show a
      minimum granularity of 30 minutes.

           MIN-GRANULARITY:PT30M
```

3.4. Maximum Granularity Component Property

```
   Property Name:  MAX-GRANULARITY

   Purpose:  This property is used to convey the maximum granularity of
      an imprecise event.

   Value Type:  DURATION

   Property Parameters:  IANA and non-standard property parameters can
      be specified on this property.

   Conformance:  This property can be specified once in a
      "VIMPRECISEEVENT".

   Description:  This property defines the maximum duration that an
      event to be scheduled in order to accomplish a liquid task
      described by an imprecise event must have.
```

Format Definition: This property is defined by the following notation:

```
max-granularity  = "MAX-GRANULARITY" max-granulparam ":"
                   dur-value CRLF

max-granulparam  = *(";" other-param)
```

Example: The following is an example of this property to show a maximum granularity of 2 hours.

```
MAX-GRANULARITY:PT2H
```

3.5. Rank Component Property

Property Name: RANK

Purpose: This property is used to convey the percent rank for regular and imprecise event components on the context of alternative events, and for availability components on the context of an imprecise event.

Value Type: INTEGER

Property Parameters: IANA and non-standard property parameters can be specified on this property.

Conformance: This property can be specified once in a "VEVENT" or "VIMPRECISEEVENT" calendar component grouped by a "VALTERNATIVEEVENTS" calendar component. This property can be specified once in an "AVAILABLE" sub-component of a "VAVAILABILITY" component.

Description: The property value is a positive integer between zero and one hundred. A value of "0" indicates the lowest possible rank. A value of "100" indicates the highest possible rank. The rank may express the preference or relative availability of the calendar user regarding the event description or available periods of time it refers to.

Format Definition: This property is defined by the following notation:

```
rank       = "RANK" rankparam ":" integer CRLF

rankparam  = *(";" other-param)
```

Example: The following is an example of this property to show a 40% rank.

```
RANK:40
```

3.6. Rank Property Parameter

Parameter Name: RANK

Purpose: To convey a percent rank for the property value.

Format Definition: This property parameter is defined by the following notation:

```
rankpparam  = "RANK" "=" integer
```

Description: The parameter value is a positive integer between zero and one hundred. A value of "0" indicates the lowest possible rank. A value of "100" indicates the highest possible rank. This parameter can be used on the "FREEBUSY" property to indicate the rank of the period of time described by it.

Example: The following is an example of this parameter to represent a 40% rank for an interval specified by a "FREEBUSY" property.

```
FREEBUSY;FBTYPE=FREE;RANK=40:20080401T160000Z/PT1H
```

4. iTIP Extensions

The table below shows various combinations of calendar components and the method types that this memo supports.

	VIMPRECISEEVENT	VALTERNATIVEEVENTS
PUBLISH	Yes	Yes
REQUEST	Yes	Yes
REPLY	Yes	Yes
ADD	No	No
CANCEL	Yes	Yes
REFRESH	Yes	Yes
COUNTER	Yes	Yes
DECLINECOUNTER	Yes	Yes

Implementations of this memo MUST support interchangeable use of

VEVENT, VIMPRECISEEVENT and VALTERNATIVEEVENTS components that correspond to a same event. This applies for the PUBLISH, REQUEST and COUNTER methods, as described below.

PUBLISH: Any of the event components can be interchangeably updated by an "Organizer" with a PUBLISH method. For instance, a VIMPRECISEEVENT can be replaced by a VALTERNATIVEEVENTS description that corresponds to an updated version of the event.

REQUEST: Any of the event components can be interchangeably updated by an "Organizer" with a REQUEST method. For instance, a REFRESH on an obsolete VIMPRECISEEVENT component can return a VEVENT component that corresponds to an updated version of the event.

COUNTER: Any of the event components can be interchangeably countered by an "Attendee" with a COUNTER method. For instance, a VEVENT can be used to COUNTER a VIMPRECISEEVENT proposal.

To be interpreted as different representations of the same event, VEVENT, VIMPRECISEEVENT and VALTERNATIVEEVENTS components MUST keep the original UID, updating the SEQUENCE property accordingly with the same rules described on [I-D.ietf-calsify-2446bis].

The REPLY, CANCEL, REFRESH and DECLINECOUNTER methods MUST NOT be issued interchangeably, i.e., they can be used only with the same component type. For instance, a VIMPRECISEEVENT cannot be cancelled by using a VEVENT CANCEL method with the same UID.

Each method type is defined in terms of its associated components and properties. Some components and properties are required, some are optional and others are excluded. The restrictions are expressed in this document using a simple "restriction table", as defined in [I-D.ietf-calsify-2446bis].

4.1. Methods for VIMPRECISEEEVENT Calendar Components

This section defines the property set restrictions for the method types that are applicable to the "VIMPRECISEEEVENT" calendar component. Each method is defined using a table that clarifies the property constraints that define the particular method.

The following summarizes the methods that are defined for the "VIMPRECISEEEVENT" calendar component.

Method	Description

PUBLISH	Post notification of an imprecise event. Used primarily as a method of advertising the existence of an imprecise event.
REQUEST	Make a request for an imprecise event. This is an explicit invitation to one or more "Attendees". Imprecise event requests are also used to update or change an existing event. Clients that cannot handle REQUEST may degrade the event to view it as a PUBLISH.
REPLY	Reply to an imprecise event request. Clients may set their status ("partstat") to ACCEPTED, DECLINED, TENTATIVE, or DELEGATED.
CANCEL	Cancel an existing imprecise event.
REFRESH	A request is sent to an "Organizer" by an "Attendee" asking for the latest version of an imprecise event to be resent to the requester.
COUNTER	Counter a REQUEST with an alternative proposal, Sent by an "Attendee" to the "Organizer".
DECLINECOUNTER	Decline a counter proposal. Sent to an "Attendee" by the "Organizer".

4.1.1. PUBLISH

The "PUBLISH" method in a "VIMPRECISEEEVENT" calendar component is an unsolicited posting of an iCalendar object. Any CU may add published components to their calendar. The "Organizer" MUST be present in a published iCalendar component. "Attendees" MUST NOT be present. Its expected usage is for encapsulating an arbitrary imprecise event as an iCalendar object. The "Organizer" may subsequently update (with another "PUBLISH" method) or cancel (with a "CANCEL" method) a previously published "VIMPRECISEEVENT" calendar component.

This method type is an iCalendar object that conforms to the following property constraints:

Component/Property	Presence	Comment
METHOD	1	MUST equal "PUBLISH"

VIMPRECISEEVENT	1	
DTSTAMP	1	
ORGANIZER	1	
SUMMARY	1	Can be null
UID	1	
SEQUENCE	0 or 1	MUST be present if value is greater than 0, MAY be present if 0
ATTACH	0+	
CATEGORIES	0+	
CLASS	0 or 1	
COMMENT	0+	
CONTACT	0+	
CREATED	0 or 1	
DESCRIPTION	0 or 1	
DUE	0 or 1	
DURATION	0 or 1	MUST be present if PERCENT is present
GEO	0 or 1	
LAST-MODIFIED	0 or 1	
LOCATION	0 or 1	
MIN-GRANULARITY	0 or 1	if MAX-GRANULARITY is present, SHOULD be less than it
MAX-GRANULARITY	0 or 1	if MIN-GRANULARITY is present, SHOULD be more than it
PERCENT	0 or 1	
PRIORITY	0 or 1	
RELATED-TO	0+	
RESOURCES	0+	
RRULE	0+	
STATUS	0 or 1	MAY be one of TENTATIVE/CONFIRMED/CANCELLED
TRANSP	0 or 1	
URL	0 or 1	
X-PROPERTY	0+	
ATTENDEE	0	
RANK	0	
REQUEST-STATUS	0	
VAVAILABILITY	0+	
UID	1	
BUSYTYPE	0 or 1	
CATEGORIES	0+	
COMMENT	0+	
CONTACT	0+	
CREATED	0 or 1	
DTSTAMP	0 or 1	

```
|     DTSTART          | 0 or 1   |                                    |
|     DTEND            | 0 or 1   | if present, DURATION MUST NOT      |
|                      |          | occur                              |
|     DURATION         | 0 or 1   | if present, DTEND MUST NOT occur   |
|     LAST-MODIFIED    | 0 or 1   |                                    |
|     ORGANIZER        | 0 or 1   | MUST contain the address of        |
|                      |          | originator of available time       |
|                      |          | data                               |
|     SUMMARY          | 0 or 1   |                                    |
|     URL              | 0 or 1   | Specifies available time URL       |
|     X-PROPERTY       | 0+       |                                    |
|     SEQUENCE         | 0        |                                    |
|                      |          |                                    |
|     AVAILABLE        | 1+       |                                    |
|       UID            | 1        |                                    |
|       DTSTART        | 1        |                                    |
|       DTEND          | 0 or 1   | if present, DURATION MUST NOT      |
|                      |          | occur                              |
|       DURATION       | 0 or 1   | if present, DTEND MUST NOT occur   |
|       DTSTAMP        | 0 or 1   |                                    |
|       CATEGORIES     | 0+       |                                    |
|       COMMENT        | 0+       |                                    |
|       CONTACT        | 0+       |                                    |
|       CREATED        | 0 or 1   |                                    |
|       EXDATE         | 0+       |                                    |
|       LAST-MODIFIED  | 0 or 1   |                                    |
|       RANK           | 0 or 1   |                                    |
|       RDATE          | 0+       |                                    |
|       RECURRENCE-ID  | 0 or 1   |                                    |
|       RRULE          | 0 or 1   |                                    |
|       SUMMARY        | 0 or 1   |                                    |
|       X-PROPERTY     | 0+       |                                    |
|                      |          |                                    |
|   VFREEBUSY          | 0+       |                                    |
|     DTSTART          | 1        | DateTime values must be in UTC     |
|     DTEND            | 1        | DateTime values must be in UTC     |
|     FREEBUSY         | 1+       | MUST NOT be "BUSY-UNAVAILABLE"     |
|                      |          | time.  Multiple instances are      |
|                      |          | allowed.  Multiple instances       |
|                      |          | must be sorted in ascending        |
|                      |          | order                              |
|     ORGANIZER        | 1        | MUST contain the address of        |
|                      |          | originator of free time data       |
|     UID              | 1        |                                    |
|     COMMENT          | 0+       |                                    |
|     CONTACT          | 0 or 1   |                                    |
|     DTSTAMP          | 0 or 1   |                                    |
```

	URL	0 or 1	Specifies free time URL
	X-PROPERTY	0+	
	ATTENDEE	0	
	DURATION	0	
	REQUEST-STATUS	0	
VTIMEZONE		0+	MUST be present if any date/time refers to a timezone
X-COMPONENT		0+	
VALARM		0	
VALTERNATIVEVENTS		0	
VAVAILABILITY		0	
VEVENT		0	
VFREEBUSY		0	
VJOURNAL		0	
VTODO		0	

4.1.2. REQUEST

The "REQUEST" method in a "VIMPRECISEEEVENT" calendar component provides the following scheduling negotiation functions:

- Invite "Attendees" to an imprecise event
- Response to a REFRESH request
- Update the details of an existing imprecise event
- Update the status of "Attendees" of an existing imprecise event
- Reconfirm an existing imprecise event
- Forward a "VIMPRECISEEEVENT" to another uninvited CU
- For an existing "VIMPRECISEEEVENT" calendar component, delegate the role of "Attendee" to another CU
- For an existing "VIMPRECISEEEVENT" calendar component, changing the role of "Organizer" to another CU

This method type is an iCalendar object that conforms to the following property constraints:

Component/Property	Presence	Comment

METHOD	1	MUST equal "REQUEST"
VIMPRECISEEVENT	1	
ATTENDEE	1+	
DTSTAMP	1	
ORGANIZER	1	
SEQUENCE	0 or 1	MUST be present if value is greater than 0, MAY be present if 0
SUMMARY	1	Can be null
UID	1	
ATTACH	0+	
CATEGORIES	0+	
CLASS	0 or 1	
COMMENT	0+	
CONTACT	0+	
CREATED	0 or 1	
DESCRIPTION	0 or 1	
DUE	0 or 1	
DURATION	0 or 1	MUST be present if PERCENT is present
GEO	0 or 1	
LAST-MODIFIED	0 or 1	
LOCATION	0 or 1	
MIN-GRANULARITY	0 or 1	if MAX-GRANULARITY is present, SHOULD be less than it
MAX-GRANULARITY	0 or 1	if MIN-GRANULARITY is present, SHOULD be more than it
PERCENT	0 or 1	
PRIORITY	0 or 1	
RELATED-TO	0+	
REQUEST-STATUS	0+	
RESOURCES	0+	
RRULE	0+	
STATUS	0 or 1	MAY be one of TENTATIVE/CONFIRMED
TRANSP	0 or 1	
URL	0 or 1	
X-PROPERTY	0+	
RANK	0	
VAVAILABILITY	0+	Same restrictions than those for the PUBLISH method
VFREEBUSY	0+	Same restrictions than those for the PUBLISH method

VTIMEZONE	0+	MUST be present if any date/time refers to a timezone
X-COMPONENT	0+	
VALARM	0	
VALTERNATIVEVENTS	0	
VAVAILABILITY	0	
VEVENT	0	
VFREEBUSY	0	
VJOURNAL	0	
VTODO	0	

Analogously to the "REQUEST" method in a "VEVENT" component, the use cases described in sections 3.2.2.2 to 3.2.2.7 of [I-D.ietf-calsify-2446bis] apply for the "VIMPRECISEEVENT" calendar component.

4.1.3. REPLY

The "REPLY" method in a "VIMPRECISEEEVENT" calendar component is used to respond (e.g., accept or decline) to a "REQUEST" or to reply to a delegation "REQUEST". When used to provide a delegation response, the "Delegator" SHOULD include the calendar address of the "Delegate" on the "DELEGATED-TO" property parameter of the "Delegator's" "ATTENDEE" property. The "Delegate" SHOULD include the calendar address of the "Delegator" on the "DELEGATED-FROM" property parameter of the "Delegate's" "ATTENDEE" property.

The "REPLY" method may also be used to respond to an unsuccessful "REQUEST" method.

The "Organizer" of an imprecise event may receive the "REPLY" method from a CU not in the original "REQUEST". For example, a "REPLY" may be received from a "Delegate" to an event. In addition, the "REPLY" method may be received from an unknown CU (a "Party Crasher"). This uninvited "Attendee" may be accepted, or the "Organizer" may cancel the event for the uninvited "Attendee" by sending a "CANCEL" method to the uninvited "Attendee".

An "Attendee" can include a message to the "Organizer" using the "COMMENT" property. For example, if the user indicates tentative acceptance and wants to let the "Organizer" know why, the reason can be expressed in the "COMMENT" property value.

The "Organizer" may also receive a "REPLY" from one CU on behalf of another. Like the scenario enumerated above for the "Organizer", "Attendees" may have another CU respond on their behalf. This is done using the "SENT-BY" parameter.

The optional properties listed in the table below (those listed as "0+" or "0 or 1") MUST NOT be changed from those of the original request. If property changes are desired the COUNTER message must be used.

This method type is an iCalendar object that conforms to the following property constraints:

Component/Property	Presence	Comment
METHOD	1	MUST equal "REQUEST"
VIMPRECISEEVENT	1	
ATTENDEE	1+	
DTSTAMP	1	
ORGANIZER	1	
SEQUENCE	0 or 1	MUST be present if value is greater than 0, MAY be present if 0
SUMMARY	1	Can be null
UID	1	
ATTACH	0+	
CATEGORIES	0+	
CLASS	0 or 1	
COMMENT	0+	
CONTACT	0+	
CREATED	0 or 1	
DESCRIPTION	0 or 1	
DUE	0 or 1	
DURATION	0 or 1	MUST be present if PERCENT is present
GEO	0 or 1	
LAST-MODIFIED	0 or 1	
LOCATION	0 or 1	
MIN-GRANULARITY	0 or 1	if MAX-GRANULARITY is present, SHOULD be less than it

MAX-GRANULARITY	0 or 1	if MIN-GRANULARITY is present, SHOULD be more than it
PERCENT	0 or 1	
PRIORITY	0 or 1	
RELATED-TO	0+	
REQUEST-STATUS	0+	
RESOURCES	0+	
RRULE	0+	
STATUS	0 or 1	
TRANSP	0 or 1	
URL	0 or 1	
X-PROPERTY	0+	
RANK	0	
VAVAILABILITY	0+	Same restrictions than those for the PUBLISH method
VFREEBUSY	0+	Same restrictions than those for the PUBLISH method
VTIMEZONE	0+	MUST be present if any date/time refers to a timezone
X-COMPONENT	0+	
VALARM	0	
VALTERNATIVEVENTS	0	
VAVAILABILITY	0	
VEVENT	0	
VFREEBUSY	0	
VJOURNAL	0	
VTODO	0	

4.1.4. CANCEL

The "CANCEL" method in a "VIMPRECISEEEVENT" calendar component is used to send a cancellation notice of an existing imprecise event request to the "Attendees". The message is sent by the "Organizer" of the event.

```
When a "VIMPRECISEEEVENT" is cancelled, the "SEQUENCE" property value
MUST be incremented.

This method type is an iCalendar object that conforms to the
following property constraints:

+--------------------+----------+-----------------------------------+
| Component/Property | Presence | Comment                           |
+--------------------+----------+-----------------------------------+
| METHOD             | 1        | MUST equal "CANCEL"               |
|                    |          |                                   |
| VIMPRECISEEVENT    | 1        |                                   |
|   ATTENDEE         | 1+       | MUST include all "Attendees"      |
|                    |          | being removed the event.  MUST    |
|                    |          | include all "Attendees" if the    |
|                    |          | entire event is cancelled.        |
|   DTSTAMP          | 1        |                                   |
|   ORGANIZER        | 1        |                                   |
|   SEQUENCE         | 0 or 1   | MUST be present if value is       |
|                    |          | greater than 0, MAY be present if |
|                    |          | 0                                 |
|   SUMMARY          | 1        | Can be null                       |
|   UID              | 1        | MUST be the UID of the original   |
|                    |          | REQUEST                           |
|   ATTACH           | 0+       |                                   |
|   CATEGORIES       | 0+       |                                   |
|   CLASS            | 0 or 1   |                                   |
|   COMMENT          | 0+       |                                   |
|   CONTACT          | 0+       |                                   |
|   CREATED          | 0 or 1   |                                   |
|   DESCRIPTION      | 0 or 1   |                                   |
|   DUE              | 0 or 1   |                                   |
|   DURATION         | 0 or 1   | MUST be present if PERCENT is     |
|                    |          | present                           |
|   GEO              | 0 or 1   |                                   |
|   LAST-MODIFIED    | 0 or 1   |                                   |
|   LOCATION         | 0 or 1   |                                   |
|   MIN-GRANULARITY  | 0 or 1   | if MAX-GRANULARITY is present,    |
|                    |          | SHOULD be less than it            |
|   MAX-GRANULARITY  | 0 or 1   | if MIN-GRANULARITY is present,    |
|                    |          | SHOULD be more than it            |
|   PERCENT          | 0 or 1   |                                   |
|   PRIORITY         | 0 or 1   |                                   |
|   RELATED-TO       | 0+       |                                   |
|   REQUEST-STATUS   | 0+       |                                   |
|   RESOURCES        | 0+       |                                   |
|   RRULE            | 0+       |                                   |
```

STATUS	0 or 1	MUST be set to CANCELLED. If uninviting specific "Attendees" then MUST NOT be included
TRANSP	0 or 1	
URL	0 or 1	
X-PROPERTY	0+	
RANK	0	
VAVAILABILITY	0+	Same restrictions than those for the PUBLISH method
VFREEBUSY	0+	Same restrictions than those for the PUBLISH method
VTIMEZONE	0+	MUST be present if any date/time refers to a timezone
X-COMPONENT	0+	
VALARM	0	
VALTERNATIVEVENTS	0	
VAVAILABILITY	0	
VEVENT	0	
VFREEBUSY	0	
VJOURNAL	0	
VTODO	0	

4.1.5. REFRESH

The "REFRESH" method in a "VIMPRECISEEEVENT" calendar component is used by "Attendees" of an existing imprecise event to request an updated description from the event "Organizer". The "REFRESH" method must specify the "UID" property of the event to update. The "Organizer" responds with the latest description and version of the event.

This method type is an iCalendar object that conforms to the following property constraints:

Component/Property	Presence	Comment
METHOD	1	MUST equal "REFRESH"
VIMPRECISEEVENT	1	
ATTENDEE	1	MUST be the address of requester
DTSTAMP	1	
ORGANIZER	1	
UID	1	MUST be the UID associated with
		original REQUEST
COMMENT	0+	
X-PROPERTY	0+	
ATTACH	0	
CATEGORIES	0	
CLASS	0	
CONTACT	0	
CREATED	0	
DESCRIPTION	0	
DUE	0	
DURATION	0	
GEO	0	
LAST-MODIFIED	0	
LOCATION	0	
MIN-GRANULARITY	0	
MAX-GRANULARITY	0	
PERCENT	0	
PRIORITY	0	
RANK	0	
RELATED-TO	0	
REQUEST-STATUS	0	
RESOURCES	0	
RRULE	0	
SEQUENCE	0	
STATUS	0	
SUMMARY	0	
TRANSP	0	
URL	0	
VAVAILABILITY	0	
VFREEBUSY	0	
X-COMPONENT	0+	
VALARM	0	
VALTERNATIVEEVENTS	0	

VAVAILABILITY	0	
VEVENT	0	
VJOURNAL	0	
VTIMEZONE	0	
VTODO	0	

4.1.6. COUNTER

The "COUNTER" method in a "VIMPRECISEEEVENT" calendar component is used by an "Attendee" of an existing event (a regular event, an imprecise event, or an alternatives event) to submit to the "Organizer" a counter proposal to the event description. The "Attendee" sends this message to the "Organizer" of the event.

The counter proposal is an iCalendar object consisting of a VIMPRECISEEEVENT calendar component describing the complete description of the alternate event.

The "Organizer" rejects the counter proposal by sending the "Attendee" a VIMPRECISEEEVENT "DECLINECOUNTER" method. The "Organizer" accepts the counter proposal by rescheduling the event as described in section 3.2.2.1 of [I-D.ietf-calsify-2446bis].

This method type is an iCalendar object that conforms to the following property constraints:

Component/Property	Presence	Comment
METHOD	1	MUST equal "COUNTER"
VIMPRECISEEVENT	1	
DTSTAMP	1	
ORGANIZER	1	MUST be the "Organizer" of the original event
SEQUENCE	0 or 1	MUST be present if value is greater than 0, MAY be present if 0
SUMMARY	1	Can be null
UID	1	MUST be the UID associated with the REQUEST being countered

ATTACH	0+	
ATTENDEE	0+	Can also be used to propose other "Attendees"
CATEGORIES	0+	
CLASS	0 or 1	
COMMENT	0+	
CONTACT	0+	
CREATED	0 or 1	
DESCRIPTION	0 or 1	
DUE	0 or 1	
DURATION	0 or 1	MUST be present if PERCENT is present
GEO	0 or 1	
LAST-MODIFIED	0 or 1	
LOCATION	0 or 1	
MIN-GRANULARITY	0 or 1	if MAX-GRANULARITY is present, SHOULD be less than it
MAX-GRANULARITY	0 or 1	if MIN-GRANULARITY is present, SHOULD be more than it
PERCENT	0 or 1	
PRIORITY	0 or 1	
RELATED-TO	0+	
REQUEST-STATUS	0+	
RESOURCES	0+	
RRULE	0+	
STATUS	0 or 1	Value must be one of CONFIRMED/TENATIVE/CANCELLED
TRANSP	0 or 1	
URL	0 or 1	
X-PROPERTY	0+	
RANK	0	
VAVAILABILITY	0+	Same restrictions than those for the PUBLISH method
VFREEBUSY	0+	Same restrictions than those for the PUBLISH method
VTIMEZONE	0+	MUST be present if any date/time refers to a timezone
X-COMPONENT	0+	
VALARM	0	
VALTERNATIVEVENTS	0	

VAVAILABILITY	0	
VEVENT	0	
VFREEBUSY	0	
VJOURNAL	0	
VTODO	0	

4.1.7. DECLINECOUNTER

The "DECLINECOUNTER" method in a "VIMPRECISEEEVENT" calendar component is used by the "Organizer" of an imprecise event to reject a counter proposal submitted by an "Attendee". The "Organizer" must send the "DECLINECOUNTER" message to the "Attendee" that sent the "COUNTER" method to the "Organizer".

This method type is an iCalendar object that conforms to the following property constraints:

Component/Property	Presence	Comment
METHOD	1	MUST equal "DECLINECOUNTER"
VIMPRECISEEVENT	1	
DTSTAMP	1	
ORGANIZER	1	
UID	1	MUST be same UID specified in original REQUEST and subsequent COUNTER
COMMENT	0+	
REQUEST-STATUS	0+	
SEQUENCE	0 or 1	MUST be present if value is greater than 0, MAY be present if 0
X-PROPERTY	0+	
ATTACH	0	
ATTENDEE	0	
CATEGORIES	0	
CLASS	0	
CONTACT	0	
CREATED	0	
DESCRIPTION	0	
DUE	0	

```
|   DURATION          | 0        |                                    |
|   GEO               | 0        |                                    |
|   LAST-MODIFIED     | 0        |                                    |
|   LOCATION          | 0        |                                    |
|   MIN-GRANULARITY   | 0        |                                    |
|   MAX-GRANULARITY   | 0        |                                    |
|   PERCENT           | 0        |                                    |
|   PRIORITY          | 0        |                                    |
|   RANK              | 0        |                                    |
|   RELATED-TO        | 0        |                                    |
|   RESOURCES         | 0        |                                    |
|   RRULE             | 0        |                                    |
|   STATUS            | 0        |                                    |
|   SUMMARY           | 0        |                                    |
|   TRANSP            | 0        |                                    |
|   URL               | 0        |                                    |
|                     |          |                                    |
|   VAVAILABILITY     | 0        |                                    |
|                     |          |                                    |
|   VFREEBUSY         | 0        |                                    |
|                     |          |                                    |
| X-COMPONENT         | 0+       |                                    |
|                     |          |                                    |
| VALARM              | 0        |                                    |
|                     |          |                                    |
| VALTERNATIVEEVENTS  | 0        |                                    |
|                     |          |                                    |
| VAVAILABILITY       | 0        |                                    |
|                     |          |                                    |
| VEVENT              | 0        |                                    |
|                     |          |                                    |
| VJOURNAL            | 0        |                                    |
|                     |          |                                    |
| VTODO               | 0        |                                    |
|                     |          |                                    |
| VTIMEZONE           | 0        |                                    |
+---------------------+----------+------------------------------------+
```

4.2. Methods for VALTERNATIVEEVENTS Calendar Components

This section defines the property set restrictions for the method types that are applicable to the "VALTERNATIVEEVENTS" calendar component. Each method is defined using a table that clarifies the property constraints that define the particular method.

The following summarizes the methods that are defined for the "VALTERNATIVEEVENTS" calendar component.

Method	Description
PUBLISH	Post notification of an alternatives event. Used primarily as a method of advertising the existence of an alternatives event.
REQUEST	Make a request for an alternatives event. This is an explicit invitation to one or more "Attendees". Alternatives event requests are also used to update or change an existing event. Clients that cannot handle REQUEST may degrade the event to view it as a PUBLISH.
REPLY	Reply to an alternatives event request. Clients may set their status ("partstat") to ACCEPTED, DECLINED, TENTATIVE, or DELEGATED.
CANCEL	Cancel an existing alternatives event.
REFRESH	A request is sent to an "Organizer" by an "Attendee" asking for the latest version of an alternatives event to be resent to the requester.
COUNTER	Counter a REQUEST with an alternative proposal, Sent by an "Attendee" to the "Organizer".
DECLINECOUNTER	Decline a counter proposal. Sent to an "Attendee" by the "Organizer".

4.2.1. PUBLISH

The "PUBLISH" method in a "VALTERNATIVEEVENTS" calendar component is an unsolicited posting of an iCalendar object. Any CU may add published components to their calendar. The "Organizer" MUST be present in a published iCalendar component. "Attendees" MUST NOT be present. Its expected usage is for encapsulating an arbitrary alternatives event as an iCalendar object. The "Organizer" may subsequently update (with another "PUBLISH" method) or cancel (with a "CANCEL" method) a previously published "VALTERNATIVEEVENTS" calendar component.

This method type is an iCalendar object that conforms to the following property constraints:

Component/Property	Presence	Comment
METHOD	1	MUST equal "PUBLISH"
VALTERNATIVEEVENTS	1	
ATTENDEE	1+	
DTSTAMP	1	
ORGANIZER	1	
SEQUENCE	0 or 1	MUST be present if value is greater than 0, MAY be present if 0
SUMMARY	1	Can be null
UID	1	
ATTACH	0+	
CATEGORIES	0+	
CLASS	0 or 1	
COMMENT	0+	
CONTACT	0+	
CREATED	0 or 1	
DESCRIPTION	0 or 1	
GEO	0 or 1	
LAST-MODIFIED	0 or 1	
LOCATION	0 or 1	
PRIORITY	0 or 1	
RELATED-TO	0+	
REQUEST-STATUS	0+	
RESOURCES	0+	
STATUS	0 or 1	MAY be one of TENTATIVE/CONFIRMED/CANCELLED
TRANSP	0 or 1	
URL	0 or 1	
X-PROPERTY	0+	
VEVENT	0+	MUST be one or more of VEVENT and VIMPRECISEEVENT.
DTSTART	1	
UID	1	Components with the same UID MUST be interpreted as a single event with instance-specific information. Components with different UID MUST be interpreted as different event alternatives.
ATTACH	0+	
ATTENDEE	0+	

CATEGORIES	0+		
CLASS	0 or 1		
COMMENT	0+		
CONTACT	0+		
CREATED	0 or 1		
DESCRIPTION	0 or 1		
DTSTAMP	0 or 1		
DTEND	0 or 1	if present DURATION MUST NOT be present	
DURATION	0 or 1	if present DTEND MUST NOT be present	
EXDATE	0+		
EXRULE	0+		
GEO	0 or 1		
LAST-MODIFIED	0 or 1		
LOCATION	0 or 1		
PRIORITY	0 or 1		
ORGANIZER	0 or 1		
RANK	0+		
RDATE	0+		
RECURRENCE-ID	0 or 1	only if referring to an instance of a recurring calendar component. Otherwise it MUST NOT be present.	
RELATED-TO	0+		
REQUEST-STATUS	0+		
RESOURCES	0+		
RRULE	0+		
STATUS	0 or 1	MAY be one of TENTATIVE/CONFIRMED/CANCELLED	
SUMMARY	0 or 1	Can be null	
TRANSP	0 or 1		
URL	0 or 1		
X-PROPERTY	0+		
SEQUENCE	0		
VIMPRECISEEVENT	0+	MUST be one or more of VEVENT and VIMPRECISEEVENT	
UID	1		
ATTACH	0+		
ATTENDEE	0+		
CATEGORIES	0+		
CLASS	0 or 1		
COMMENT	0+		
CONTACT	0+		
CREATED	0 or 1		
DESCRIPTION	0 or 1		

	DTSTAMP	0 or 1	
	DUE	0 or 1	
	DURATION	0 or 1	MUST be present if DUE is present
	GEO	0 or 1	
	LAST-MODIFIED	0 or 1	
	LOCATION	0 or 1	
	MIN-GRANULARITY	0 or 1	if MAX-GRANULARITY is present, SHOULD be less than it
	MAX-GRANULARITY	0 or 1	if MIN-GRANULARITY is present, SHOULD be more than it
	ORGANIZER	0 or 1	
	PERCENT	0 or 1	
	PRIORITY	0 or 1	
	RANK	0 or 1	
	RELATED-TO	0+	
	REQUEST-STATUS	0+	
	RESOURCES	0+	
	RRULE	0+	
	STATUS	0 or 1	MAY be one of TENTATIVE/CONFIRMED/CANCELLED
	SUMMARY	0 or 1	Can be null
	TRANSP	0 or 1	
	URL	0 or 1	
	X-PROPERTY	0+	
	SEQUENCE	0	
	VAVAILABILITY	0+	Same restrictions than those for the PUBLISH method in the VIMPRECISEEVENT component
	VFREEBUSY	0+	Same restrictions than those for the PUBLISH method in the VIMPRECISEEVENT component
VTIMEZONE		0+	MUST be present if any date/time refers to a timezone
X-COMPONENT		0+	
VALARM		0	
VAVAILABILITY		0	
VEVENT		0	
VFREEBUSY		0	

VIMPRECISEEVENT	0		
VJOURNAL	0		
VTODO	0		

4.2.2. REQUEST

The "REQUEST" method in a "VALTERNATIVEEVENTS" calendar component provides the following scheduling negotiation functions:

- Invite "Attendees" to an alternative events
- Response to a REFRESH request
- Update the details of an existing alternative events
- Update the status of "Attendees" of an existing alternative events
- Reconfirm an existing alternatives event
- Forward a "VALTERNATIVEEVENTS" to another uninvited CU
- For an existing "VALTERNATIVEEVENTS" calendar component, delegate the role of "Attendee" to another CU
- For an existing "VALTERNATIVEEVENTS" calendar component, changing the role of "Organizer" to another CU

The "Organizer" originates the "REQUEST". The recipients of the "REQUEST" method are the CUs invited to the event, the "Attendees". "Attendees" use the "REPLY" method to convey attendance status to the "Organizer".

The "UID" and "SEQUENCE" properties are used to distinguish the various uses of the "REQUEST" method. If the "UID" property value in the "REQUEST" is not found on the recipient's calendar, then the "REQUEST" is for a new "VALTERNATIVEEVENTS" calendar component. If the "UID" property value is found on the recipient's calendar, then the "REQUEST" is for an update, or a reconfirm of the "VALTERNATIVEEVENTS" calendar component.

This method type is an iCalendar object that conforms to the following property constraints:

Component/Property	Presence	Comment
METHOD	1	MUST equal "REQUEST"
VALTERNATIVEEVENTS	1	
ATTENDEE	1+	

DTSTAMP	1	
ORGANIZER	1	
SEQUENCE	0 or 1	MUST be present if value is greater than 0, MAY be present if 0
SUMMARY	1	Can be null
UID	1	
ATTACH	0+	
CATEGORIES	0+	
CLASS	0 or 1	
COMMENT	0+	
CONTACT	0+	
CREATED	0 or 1	
DESCRIPTION	0 or 1	
GEO	0 or 1	
LAST-MODIFIED	0 or 1	
LOCATION	0 or 1	
PRIORITY	0 or 1	
RELATED-TO	0+	
REQUEST-STATUS	0+	
RESOURCES	0+	
STATUS	0 or 1	MAY be one of TENTATIVE/CONFIRMED
TRANSP	0 or 1	
URL	0 or 1	
X-PROPERTY	0+	
VEVENT	0+	Same restrictions than those for the PUBLISH method
VIMPRECISEEVENT	0+	Same restrictions than those for the PUBLISH method
VTIMEZONE	0+	MUST be present if any date/time refers to a timezone
X-COMPONENT	0+	
VALARM	0	
VAVAILABILITY	0	
VEVENT	0	
VFREEBUSY	0	
VIMPRECISEEVENT	0	

VJOURNAL	0	
VTODO	0	

Analogously to the "REQUEST" method in a "VEVENT" component, the use cases described in sections 3.2.2.2 to 3.2.2.7 of [I-D.ietf-calsify-2446bis] apply for the "VALTERNATIVEEVENTS" calendar component.

4.2.3. REPLY

The "REPLY" method in a "VALTERNATIVEEVENTS" calendar component is used to respond (e.g., accept or decline) to a "REQUEST" or to reply to a delegation "REQUEST". When used to provide a delegation response, the "Delegator" SHOULD include the calendar address of the "Delegate" on the "DELEGATED-TO" property parameter of the "Delegator's" "ATTENDEE" property. The "Delegate" SHOULD include the calendar address of the "Delegator" on the "DELEGATED-FROM" property parameter of the "Delegate's" "ATTENDEE" property.

The "REPLY" method may also be used to respond to an unsuccessful "REQUEST" method.

The "Organizer" of an alternatives event may receive the "REPLY" method from a CU not in the original "REQUEST". For example, a "REPLY" may be received from a "Delegate" to an event. In addition, the "REPLY" method may be received from an unknown CU (a "Party Crasher"). This uninvited "Attendee" may be accepted, or the "Organizer" may cancel the event for the uninvited "Attendee" by sending a "CANCEL" method to the uninvited "Attendee".

An "Attendee" can include a message to the "Organizer" using the "COMMENT" property. For example, if the user indicates tentative acceptance and wants to let the "Organizer" know why, the reason can be expressed in the "COMMENT" property value.

The "Organizer" may also receive a "REPLY" from one CU on behalf of another. Like the scenario enumerated above for the "Organizer", "Attendees" may have another CU respond on their behalf. This is done using the "SENT-BY" parameter.

The optional properties listed in the table below (those listed as "0+" or "0 or 1") MUST NOT be changed from those of the original request. If property changes are desired the COUNTER message must be used.

```
This method type is an iCalendar object that conforms to the
following property constraints:

+--------------------+----------+-----------------------------------+
| Component/Property | Presence | Comment                           |
+--------------------+----------+-----------------------------------+
| METHOD             | 1        | MUST equal "REPLY"                |
|                    |          |                                   |
| VALTERNATIVEEVENTS | 1        |                                   |
|   ATTENDEE         | 1        | MUST be the address of the        |
|                    |          | Attendee replying                 |
|   DTSTAMP          | 1        |                                   |
|   ORGANIZER        | 1        |                                   |
|   SEQUENCE         | 0 or 1   | MUST if non-zero, MUST be the     |
|                    |          | sequence number of the original   |
|                    |          | REQUEST.  MAY be present if 0     |
|   SUMMARY          | 0 or 1   |                                   |
|   UID              | 1        | MUST be the UID of the original   |
|                    |          | REQUEST                           |
|   ATTACH           | 0+       |                                   |
|   CATEGORIES       | 0+       |                                   |
|   CLASS            | 0 or 1   |                                   |
|   COMMENT          | 0+       |                                   |
|   CONTACT          | 0+       |                                   |
|   CREATED          | 0 or 1   |                                   |
|   DESCRIPTION      | 0 or 1   |                                   |
|   GEO              | 0 or 1   |                                   |
|   LAST-MODIFIED    | 0 or 1   |                                   |
|   LOCATION         | 0 or 1   |                                   |
|   PRIORITY         | 0 or 1   |                                   |
|   RELATED-TO       | 0+       |                                   |
|   REQUEST-STATUS   | 0+       |                                   |
|   RESOURCES        | 0+       |                                   |
|   STATUS           | 0 or 1   |                                   |
|   TRANSP           | 0 or 1   |                                   |
|   URL              | 0 or 1   |                                   |
|   X-PROPERTY       | 0+       |                                   |
|                    |          |                                   |
|   VEVENT           | 0+       | Same restrictions than those for  |
|                    |          | the PUBLISH method                |
|                    |          |                                   |
|   VIMPRECISEEVENT  | 0+       | Same restrictions than those for  |
|                    |          | the PUBLISH method                |
|                    |          |                                   |
| VTIMEZONE          | 0+       | MUST be present if any date/time  |
|                    |          | refers to a timezone              |
|                    |          |                                   |
```

X-COMPONENT	0+	
VALARM	0	
VAVAILABILITY	0	
VEVENT	0	
VFREEBUSY	0	
VIMPRECISEEVENT	0	
VJOURNAL	0	
VTODO	0	

4.2.4. CANCEL

The "CANCEL" method in a "VALTERNATIVEEVENTS" calendar component is used to send a cancellation notice of an existing alternatives event request to the "Attendees". The message is sent by the "Organizer" of the event.

When a "VALTERNATIVEEVENTS" is cancelled, the "SEQUENCE" property value MUST be incremented.

This method type is an iCalendar object that conforms to the following property constraints:

Component/Property	Presence	Comment
METHOD	1	MUST equal "CANCEL"
VALTERNATIVEEVENTS	1	
ATTENDEE	1	MUST be the address of the Attendee replying
DTSTAMP	1	
ORGANIZER	1	
SEQUENCE	0 or 1	MUST if non-zero, MUST be the sequence number of the original REQUEST. MAY be present if 0
SUMMARY	0 or 1	
UID	1	MUST be the UID of the original REQUEST
ATTACH	0+	

CATEGORIES	0+	
CLASS	0 or 1	
COMMENT	0+	
CONTACT	0+	
CREATED	0 or 1	
DESCRIPTION	0 or 1	
GEO	0 or 1	
LAST-MODIFIED	0 or 1	
LOCATION	0 or 1	
PRIORITY	0 or 1	
RELATED-TO	0+	
REQUEST-STATUS	0+	
RESOURCES	0+	
STATUS	0 or 1	MUST be set to CANCELLED. If uninviting specific "Attendees" then MUST NOT be included.
TRANSP	0 or 1	
URL	0 or 1	
X-PROPERTY	0+	
VEVENT	0+	Same restrictions than those for the PUBLISH method
VIMPRECISEEVENT	0+	Same restrictions than those for the PUBLISH method
VTIMEZONE	0+	MUST be present if any date/time refers to a timezone
X-COMPONENT	0+	
VALARM	0	
VAVAILABILITY	0	
VEVENT	0	
VFREEBUSY	0	
VIMPRECISEEVENT	0	
VJOURNAL	0	
VTODO	0	

4.2.5. REFRESH

The "REFRESH" method in a "VALTERNATIVEEVENTS" calendar component is used by "Attendees" of an existing alternatives event to request an updated description from the event "Organizer". The "REFRESH" method must specify the "UID" property of the event to update. The "Organizer" responds with the latest description and version of the event.

This method type is an iCalendar object that conforms to the following property constraints:

Component/Property	Presence	Comment
METHOD	1	MUST equal "REFRESH"
VALTERNATIVEEVENTS	1	
ATTENDEE	1	MUST be the address of requester
DTSTAMP	1	
ORGANIZER	1	
UID	1	MUST be the UID associated with
		original REQUEST
COMMENT	0+	
X-PROPERTY	0+	
ATTACH	0	
CATEGORIES	0	
CLASS	0	
CONTACT	0	
CREATED	0	
DESCRIPTION	0	
GEO	0	
LAST-MODIFIED	0	
LOCATION	0	
PRIORITY	0	
RELATED-TO	0	
REQUEST-STATUS	0	
RESOURCES	0	
SEQUENCE	0	
STATUS	0	
SUMMARY	0	
TRANSP	0	
URL	0	
VEVENT	0	
VIMPRECISEEVENT	0	

X-COMPONENT	0+	
VALARM	0	
VAVAILABILITY	0	
VEVENT	0	
VFREEBUSY	0	
VIMPRECISEEVENT	0	
VJOURNAL	0	
VTIMEZONE	0	
VTODO	0	

4.2.6. COUNTER

The "COUNTER" method in a "VALTERNATIVEEVENTS" calendar component is used by an "Attendee" of an existing event (a regular event, an imprecise event, or an alternatives event) to submit to the "Organizer" a counter proposal to the event description. The "Attendee" sends this message to the "Organizer" of the event.

The counter proposal is an iCalendar object consisting of a VALTERNATIVEEVENTS calendar component describing the complete description of the alternate event.

The "Organizer" rejects the counter proposal by sending the "Attendee" a VALTERNATIVEEVENTS "DECLINECOUNTER" method. The "Organizer" accepts the counter proposal by rescheduling the event as described in section 3.2.2.1 of [I-D.ietf-calsify-2446bis].

This method type is an iCalendar object that conforms to the following property constraints:

Component/Property	Presence	Comment
METHOD	1	MUST equal "COUNTER"
VALTERNATIVEEVENTS	1	
ATTENDEE	1+	
DTSTAMP	1	

ORGANIZER	1	
SEQUENCE	0 or 1	MUST be present if value is greater than 0, MAY be present if 0
SUMMARY	1	Can be null
UID	1	MUST be the UID associated with original REQUEST
ATTACH	0+	
CATEGORIES	0+	
CLASS	0 or 1	
COMMENT	0+	
CONTACT	0+	
CREATED	0 or 1	
DESCRIPTION	0 or 1	
GEO	0 or 1	
LAST-MODIFIED	0 or 1	
LOCATION	0 or 1	
PRIORITY	0 or 1	
RELATED-TO	0+	
REQUEST-STATUS	0+	
RESOURCES	0+	
STATUS	0 or 1	Value must be one of CONFIRMED/TENTATIVE/CANCELLED
TRANSP	0 or 1	
URL	0 or 1	
X-PROPERTY	0+	
VEVENT	0+	Same restrictions than those for the PUBLISH method
VIMPRECISEEVENT	0+	Same restrictions than those for the PUBLISH method
VTIMEZONE	0+	MUST be present if any date/time refers to a timezone
X-COMPONENT	0+	
VALARM	0	
VAVAILABILITY	0	
VEVENT	0	
VFREEBUSY	0	
VIMPRECISEEVENT	0	

```
|                    |          |                                    |
| VJOURNAL           | 0        |                                    |
|                    |          |                                    |
| VTODO              | 0        |                                    |
+--------------------+----------+------------------------------------+
```

4.2.7. DECLINECOUNTER

The "DECLINECOUNTER" method in a "VALTERNATIVEEVENTS" calendar component is used by the "Organizer" of an alternative events to reject a counter proposal submitted by an "Attendee". The "Organizer" must send the "DECLINECOUNTER" message to the "Attendee" that sent the "COUNTER" method to the "Organizer".

This method type is an iCalendar object that conforms to the following property constraints:

```
+--------------------+----------+------------------------------------+
| Component/Property | Presence | Comment                            |
+--------------------+----------+------------------------------------+
| METHOD             | 1        | MUST equal "DECLINECOUNTER"        |
|                    |          |                                    |
| VALTERNATIVEEVENTS | 1        |                                    |
|   ATTENDEE         | 1+       |                                    |
|   DTSTAMP          | 1        |                                    |
|   ORGANIZER        | 1        |                                    |
|   UID              | 1        | MUST be same UID specified in      |
|                    |          | original REQUEST and subsequent    |
|                    |          | COUNTER                            |
|   COMMENT          | 0+       |                                    |
|   REQUEST-STATUS   | 0+       |                                    |
|   SEQUENCE         | 0 or 1   | MUST be present if value is        |
|                    |          | greater than 0, MAY be present if  |
|                    |          | 0                                  |
|   X-PROPERTY       | 0+       |                                    |
|   ATTACH           | 0        |                                    |
|   CATEGORIES       | 0        |                                    |
|   CLASS            | 0        |                                    |
|   CONTACT          | 0        |                                    |
|   CREATED          | 0        |                                    |
|   DESCRIPTION      | 0        |                                    |
|   GEO              | 0        |                                    |
|   LAST-MODIFIED    | 0        |                                    |
|   LOCATION         | 0        |                                    |
|   PRIORITY         | 0        |                                    |
|   RELATED-TO       | 0        |                                    |
|   RESOURCES        | 0        |                                    |
```

STATUS	0	
SUMMARY	0	
TRANSP	0	
URL	0	
VEVENT	0	
VIMPRECISEEVENT	0	
VTIMEZONE	0+	MUST be present if any date/time refers to a timezone
X-COMPONENT	0+	
VALARM	0	
VAVAILABILITY	0	
VEVENT	0	
VFREEBUSY	0	
VIMPRECISEEVENT	0	
VJOURNAL	0	
VTODO	0	

5. Examples

These examples are complementary to those presented in [I-D.ietf-calsify-2446bis].

5.1. Published Imprecise Event

The iCalendar object below describes a published imprecise event, organized by a person "A", with the purpose to announce an event for which the exact start time is not defined yet.

```
BEGIN:VCALENDAR
PRODID:-//Unicamp/iScheduler//EN
METHOD:PUBLISH
VERSION:2.0
BEGIN:VIMPRECISEEVENT
UID:20071005T133225Z-00001@example.com
```

```
ORGANIZER:mailto:A@example.com
DTSTAMP:20071005T133225Z
SUMMARY:Department meeting
DURATION:PT2H
BEGIN:VFREEBUSY
UID:20071005T133225Z-00001A@example.com
DTSTAMP:20071005T133225Z
FREEBUSY;FBTYPE=FREE:20080401T080000Z/20080401T120000Z
FREEBUSY;FBTYPE=FREE:20080401T140000Z/20080401T180000Z
END:VFREEBUSY
END:VIMPRECISEEVENT
END:VCALENDAR
```

5.2. An Imprecise Event Request

The iCalendar object below describes a new version of the previous event, for which "B", "C" and "D" are invited to participate. The inclusion of "E" as a non-participant is the equivalent of a 'Cc' (carbon-copy) operation in email.

```
BEGIN:VCALENDAR
PRODID:-//Unicamp/iScheduler//EN
METHOD:REQUEST
VERSION:2.0
BEGIN:VIMPRECISEEVENT
UID:20071005T133225Z-00001@example.com
SEQUENCE:1
ORGANIZER:mailto:A@example.com
ATTENDEE;ROLE=CHAIR;PARTSTAT=ACCEPTED;CN=A:mailto:A@example.com
ATTENDEE;RSVP=TRUE;TYPE=INDIVIDUAL;CN=B:mailto:B@example.com
ATTENDEE;RSVP=TRUE;TYPE=INDIVIDUAL;CN=C:mailto:C@example.com
ATTENDEE;RSVP=TRUE;TYPE=INDIVIDUAL;CN=D:mailto:D@example.com
ATTENDEE;RSVP=FALSE;TYPE=ROOM:conf_Big@example.com
ATTENDEE;ROLE=NON-PARTICIPANT;RSVP=FALSE:mailto:E@example.com
DTSTAMP:20071005T133225Z
SUMMARY:Department meeting
DURATION:PT2H
BEGIN:VFREEBUSY
UID:20071005T133225Z-00001A@example.com
FREEBUSY;FBTYPE=FREE:20080401T080000Z/20080401T120000Z
END:VFREEBUSY
END:VIMPRECISEEVENT
END:VCALENDAR
```

5.3. Reply to an Imprecise Event Request

The iCalendar object below describes an answer from "B", who accepts

the event proposal from "A".

```
BEGIN:VCALENDAR
PRODID:-//Unicamp/iScheduler//EN
METHOD:REPLY
VERSION:2.0
BEGIN:VIMPRECISEEVENT
UID:20071005T133225Z-00001@example.com
SEQUENCE:1
ORGANIZER:mailto:A@example.com
ATTENDEE;PARTSTAT=ACCEPTED;CN=B:mailto:B@example.com
REQUEST-STATUS:2.0;Success
DTSTAMP:20071011T145000Z
COMMENT:Perfect! I'll be there.
END:VIMPRECISEEVENT
END:VCALENDAR
```

5.4. Counter an Imprecise Event Request

The iCalendar object below describes a counterproposal from "C", who offers more than one option for scheduling the event. The VALTERNATIVEEVENTS component refers to the same UID of the original VIMPRECISEEVENT REQUEST.

```
BEGIN:VCALENDAR
PRODID:-//Unicamp/iScheduler//EN
METHOD:COUNTER
VERSION:2.0
BEGIN:VALTERNATIVEEVENTS
UID:20071005T133225Z-00001@example.com
SEQUENCE:1
ORGANIZER:mailto:A@example.com
ATTENDEE;ROLE=CHAIR;CN=A:mailto:A@example.com
ATTENDEE;RSVP=TRUE;TYPE=INDIVIDUAL;CN=B:mailto:B@example.com
ATTENDEE;TYPE=INDIVIDUAL;CN=C:mailto:C@example.com
ATTENDEE;RSVP=TRUE;TYPE=INDIVIDUAL;CN=D:mailto:D@example.com
ATTENDEE;RSVP=FALSE;TYPE=ROOM:conf_Big@example.com
ATTENDEE;ROLE=NON-PARTICIPANT;RSVP=FALSE:mailto:E@example.com
DTSTAMP:20071009T201659Z
SUMMARY:Department meeting
COMMENT:What do you think about these other times?
BEGIN:VEVENT
UID:20071005T133225Z-00001-A@example.com
DTSTART:20080401T090000Z
DTEND:20080401T110000Z
END:VEVENT
BEGIN:VIMPRECISEEVENT
```

```
UID:20071005T133225Z-00001-B@example.com
BEGIN:VFREEBUSY
UID:20071005T133225Z-00001-B1@example.com
FREEBUSY;FBTYPE=FREE:20080401T150000Z/20080401T190000Z
END:VFREEBUSY
END:VIMPRECISEEVENT
END:VALTERNATIVEEVENTS
END:VCALENDAR
```

5.5. Update an Event Description

The iCalendar object below describes a message from "D" asking for the updated event description. Note that it is issued with the VIMPRECISEEVENT component, which is the most recent known by "D".

```
BEGIN:VCALENDAR
PRODID:-//Unicamp/iScheduler//EN
METHOD:REFRESH
VERSION:2.0
BEGIN:VIMPRECISEEVENT
UID:20071005T133225Z-00001@example.com
SEQUENCE:1
ORGANIZER:mailto:A@example.com
ATTENDEE;CN=D:mailto:D@example.com
DTSTAMP:20071010T072859Z
END:VIMPRECISEEVENT
END:VCALENDAR
```

The iCalendar object below describes the updated version of the event, in response to "D". It reflects the acceptance of the counterproposal from "C".

```
BEGIN:VCALENDAR
PRODID:-//Unicamp/iScheduler//EN
METHOD:REQUEST
VERSION:2.0
BEGIN:VALTERNATIVEEVENTS
UID:20071005T133225Z-00001@example.com
SEQUENCE:2
ORGANIZER:mailto:A@example.com
ATTENDEE;ROLE=CHAIR;CN=A:mailto:A@example.com
ATTENDEE;RSVP=TRUE;TYPE=INDIVIDUAL;CN=B:mailto:B@example.com
ATTENDEE;RSVP=TRUE;TYPE=INDIVIDUAL;CN=C:mailto:C@example.com
ATTENDEE;RSVP=TRUE;TYPE=INDIVIDUAL;CN=D:mailto:D@example.com
ATTENDEE;RSVP=FALSE;TYPE=ROOM:conf_Big@example.com
ATTENDEE;ROLE=NON-PARTICIPANT;RSVP=FALSE:mailto:E@example.com
DTSTAMP:20071011T221500Z
```

```
SUMMARY:Department meeting
COMMENT:C's suggestion is great!
BEGIN:VEVENT
UID:20071005T133225Z-00001-A@example.com
DTSTART:20080401T090000Z
DTEND:20080401T110000Z
END:VEVENT
BEGIN:VIMPRECISEEVENT
UID:20071005T133225Z-00001-B@example.com
BEGIN:VFREEBUSY
UID:20071005T133225Z-00001-B1@example.com
FREEBUSY;FBTYPE=FREE:20080401T150000Z/20080401T190000Z
END:VFREEBUSY
END:VIMPRECISEEVENT
END:VALTERNATIVEEVENTS
END:VCALENDAR
```

6. IANA Considerations

TBD according to IANA registration procedures defined in [I-D.ietf-calsify-rfc2445bis].

7. Security Considerations

This specification does not add any additional security issues that are not already present in [I-D.ietf-calsify-rfc2445bis] and [I-D.ietf-calsify-2446bis].

8. Acknowledgements

The authors would like to thank the following individuals for their previous work on calendaring and scheduling standards, contributing ideas and support for writing this specification: Bernard Desruisseaux, Cyrus Daboo, Lisa Dusseault, and Tim Hare.

9. References

9.1. Normative References

[I-D.daboo-calendar-availability]
Daboo, C. and B. Desruisseaux, "Calendar Availability", draft-daboo-calendar-availability-00 (work in progress), November 2006.

[I-D.desruisseaux-caldav-sched]
Daboo, C., Desruisseaux, B., and L. Dusseault, "CalDAV Scheduling Extensions to WebDAV", draft-desruisseaux-caldav-sched-04 (work in progress), November 2007.

[I-D.ietf-calsify-2446bis]
Daboo, C., "iCalendar Transport-Independent Interoperability Protocol (iTIP)", draft-ietf-calsify-2446bis-04 (work in progress), November 2007.

[I-D.ietf-calsify-rfc2445bis]
Desruisseaux, B., "Internet Calendaring and Scheduling Core Object Specification (iCalendar)", draft-ietf-calsify-rfc2445bis-07 (work in progress), July 2007.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC4791] Daboo, C., Desruisseaux, B., and L. Dusseault, "Calendaring Extensions to WebDAV (CalDAV)", RFC 4791, March 2007.

9.2. Informative References

[SD98] Sen, S. and E. Durfee, "A Formal Study on Distributed Meeting Scheduling", Group Decision and Negotiation, 7(3): 265-289, 1998.

[SL02] Craig, S. and R. Letsinger, "An ontology for publishing and scheduling events and the lessons learned in developing it", HP Labs Technical Report HPL-2002-162, 2002.

Appendix A. Design Considerations

This section points out some protocol design issues concerning the described extensions.

In particular, the need of new special "event" components to support imprecise and alternative events can be questioned. They can represent an interoperability problem with current implementations, and may overload workflow scheduling requirements, making it more

complex, as VIMPRECISEEVENT and VALTERNATIVEEVENTS are intended to be used interchangeably with VEVENT during scheduling negotiation. When receiving a VEVENT / VIMPRECISEEVENT / VALTERNATIVEEVENTS scheduling message, it will be necessary to look for pre-existing components of these three types (and not only VEVENT components anymore).

One may argue that a simpler solution for this would be an extension of the existing VEVENT component and method types to support imprecise and alternative descriptions. Design choices that could be made to do so - and reasons for not adopting them - are presented here.

A.1. Avoiding VALTERNATIVEEVENTS

An extension of VEVENT to support alternative events descriptions could consist on using the RELATED-TO property to link multiple event components with each their own alternatives and creating a new property to denote that these components should be interpreted as alternative descriptions. However, because only one UID is used for all VEVENT components, that wouldn't allow for this use case: user A suggests an event with instance specific information, and user B wants to COUNTER with two alternatives - a new one (possibly with a high rank), and the original one, composed by several instances (with a lower rank). In this case, both alternatives proposed by user B must be identified by the same original UID from A's invitation. This can be solved by a VALTERNATIVEEVENTS wrapper component which would have that UID.

Using VEVENT components with different UIDs and grouping those which have the same UID as recurring events with instance specific information would not be a solution, because the reference to the UID of the original request is lost. Using another new relationship parameter and RECURRENCE-ID as a secondary key to do this grouping is also limiting, because it would prohibit two alternatives starting at the same time but with different properties (e.g., location). Thus, a solution without VALTERNATIVEEVENTS would not be really flexible.

A.2. Avoiding VIMPRECISEEVENT

Some alternatives for not creating a new VIMPRECISEEVENT component could also be considered. Relax VEVENT and allow it to be a container for VFREEBUSY and VAVAILABILITY components would be a bad solution, as present implementations would have problems attempting to parse these (so far invalid) VEVENT components. But a new property could be created to denote that the event should be interpreted as an imprecise description, and this event component could be linked with VFREEBUSY and VAVAILABILITY components via a new

property or even RELATED-TO.

But that would imply on ignoring the DTSTART and DTEND properties, because they do not make sense in an imprecise event. Since they are always expected, their meaning could be changed by the presence of another property, which should also enforce the additional recurrence limitations described on this specification. The drawback with this approach is that, by definition, an event takes up time on a calendar, what would demand workarounds (e.g., setting TRANSP: TRANSPARENT) so as to prevent these events in a calendar store to be accounted in free-busy queries by current implementations.

Using 0+ VFREEBUSY and VAVAILABILITY components in a VEVENT iTIP message would not be a satisfactory option either: if two or more imprecise descriptions were used as alternative events, the available periods of time would refer to all descriptions, and not to a specific one, due to the single UID constraint for VEVENT methods. That can be a problem if the availability depends on properties of the event description (e.g., location), what makes of VIMPRECISEEVENT a simpler way to separate semantics between regular and imprecise events.

A.3. Compatibility

We definitely agree that creating the new components breaks compatibility with current implementations and can raise interoperability issues. On the other hand, it should be considered that the purpose of these extensions is to add new semantics to scheduling negotiation, not only additional information to the concept of events as is.

While adapting the VEVENT component could be a good fallback for CUAs not implementing the extensions, it would result in meaningless - or even incorrect - interpretation, what would not be useful and might cause problems. CUAs without the capability to understand the concept of imprecise and alternative events should indeed complain about receiving an invalid iTIP message (maybe replying with a request-status error 3.12/3.13) and be out of scheduling negotiation rounds until only one VEVENT description comes into consideration.

Appendix B. Change History (to be removed by RFC Editor prior to publication)

Changes from -00:

a. Description of the methods for the new components revised and

completed.

b. Added text to clarify relationship among regular, imprecise and alternative events, and a description on how their methods can be used interchangeably.

c. Changed intended status to experimental.

d. Added percent property to imprecise events.

e. New minimum and maximum granularity properties added to imprecise events.

f. Simplified restriction tables by referring to repeating restrictions on sub-components already presented.

g. Design Considerations on appendix A rewritten and made permanent.

h. Minor editorial changes.

Authors' Addresses

Fabio Silva
University of Campinas - Institute of Computing
Caixa Postal 6176
Campinas, SP 13084-971
Brazil

Email: fabio.silva@gmail.com

Rogerio Drummond
University of Campinas - Institute of Computing
Caixa Postal 6176
Campinas, SP 13084-971
Brazil

Email: rog@ic.unicamp.br

Full Copyright Statement

Intellectual Property

The IETF takes no position regarding the validity or scope of any Intellectual Property Rights or other rights that might be claimed to pertain to the implementation or use of the technology described in this document or the extent to which any license under such rights might or might not be available; nor does it represent that it has made any independent effort to identify any such rights. Information on the procedures with respect to rights in RFC documents can be found in BCP 78 and BCP 79.

Copies of IPR disclosures made to the IETF Secretariat and any assurances of licenses to be made available, or the result of an attempt made to obtain a general license or permission for the use of such proprietary rights by implementers or users of this specification can be obtained from the IETF on-line IPR repository at http://www.ietf.org/ipr.

The IETF invites any interested party to bring to its attention any copyrights, patents or patent applications, or other proprietary rights that may cover technology that may be required to implement this standard. Please address the information to the IETF at ietf-ipr@ietf.org.

Acknowledgment

Funding for the RFC Editor function is provided by the IETF Administrative Support Activity (IASA).

Apêndice B

Detalhamento técnico das implementações

Neste apêndice, apresentamos detalhes adicionais a respeito das implementações realizadas, bem como algumas características arquiteturais das bibliotecas utilizadas que tiveram impacto no resultado final. Estas descrições utilizam terminologia comum a desenvolvedores da plataforma Java[1].

As implementações foram organizadas em uma hierarquia de classes composta por dois pacotes principais: `ischeduler.icalendar` e `ischeduler.tr`. O primeiro contém a biblioteca de entrada e saída para o padrão iCalendar com suporte aos componentes, propriedades e parâmetros que fazem parte das extensões propostas. O segundo contém a implementação interativa para TR. A figura B.1, no final do apêndice, ilustra a estrutura de classes criada no Eclipse[2], o ambiente de desenvolvimento utilizado.

B.1 Extensões do padrão iCalendar

A biblioteca iCal4j, que foi utilizada para implementar as extensões do padrão iCalendar, é formada pelos pacotes descritos a seguir.

- `net.fortuna.ical4j.data`: contém as classes responsáveis pela entrada e saída de dados, pelo *unmarshalling* de componentes iCalendar, e pelo tratamento de exceções de validação. As duas mais importantes são `CalendarBuilder`, que transforma objetos iCalendar em uma hierarquia de objetos Java correspondentes aos componentes à medida em que as tags `BEGIN:` e `END:` são encontradas, e `CalendarOutputter`, que faz o *marshalling* de objetos iCalendar, escrevendo-os em um *stream*.

[1] `http://java.sun.com/`.
[2] `http://www.eclipse.org/`.

- `net.fortuna.ical4j.filter`: contém as classes que implementam as regras básicas de validação de componentes, propriedades e parâmetros.

- `net.fortuna.ical4j.model`: além de conter as classes que representam os tipos de dados especificados no padrão iCalendar, inclui os seguintes pacotes:

  - `net.fortuna.ical4j.model.component`: contém as classes que representam os componentes do padrão iCalendar, sendo que para cada componente existe uma classe correspondente (e.g., `VEvent`);
  - `net.fortuna.ical4j.model.parameter`: contém as classes que representam os parâmetros do padrão iCalendar, sendo que para cada parâmetro existe uma classe correspondente (e.g., `Rsvp`); e
  - `net.fortuna.ical4j.model.property`: contém as classes que representam as propriedades do padrão iCalendar, sendo que para cada propriedade existe uma classe correspondente (e.g., `DtStart`).

- `net.fortuna.ical4j.transform`: pacote incipiente para o agrupamento de classes que transformam objetos iCalendar para a execução de métodos iTIP. Contém uma classe abstrata `Transformer` e uma classe `PublishTransformer`, que a estende, cuja função é incorporar o valor da propriedade `METHOD` para o método `PUBLISH`, incrementando o valor de `SEQUENCE` se necessário.

- `net.fortuna.ical4j.util`: contém classes com funções úteis para os demais pacotes (e.g., `UidGenerator`, que gera valores para a propriedade `UID`).

O pacote `ischeduler.icalendar` foi organizado com estrutura similar. Basicamente, o ponto de entrada da biblioteca estendida é uma nova implementação da classe de *marshalling* `CalendarBuilder`.

- `ischeduler.icalendar.data`: contém uma única classe, `CalendarBuilder`, que reimplementa a classe iCal4j correspondente. Ela tem o papel de interpretar os novos componentes `VIMPRECISEEVENT` e `VALTERNATIVEEVENTS` sem exigir que eles sejam X-tokens, por não fazerem parte do padrão iCalendar.

- `ischeduler.icalendar.model`: contém classes auxiliares necessárias ao conteúdo dos seguintes pacotes:

  - `ischeduler.icalendar.model.component`: contém as classes `VAlternative Events` e `VImpreciseEvent`, que correspondem aos novos componentes propostos;

  - `ischeduler.icalendar.model.parameter`: contém a classe `Rank`, que corresponde ao novo parâmetro proposto; e
  - `ischeduler.icalendar.model.property`: contém as classes `MinGranularity`, `MaxGranularity` e `Rank`, que correspondem às novas propriedades propostas, e uma classe `FreeBusy` que reimplementa a classe iCal4j correspondente, para que o parâmetro `RANK` possa ser utilizado na propriedade `FREEBUSY` sem obedecer ao padrão de X-tokens.

- `ischeduler.icalendar.util`: contém uma classe `CalendarFactory` que cria e retorna um objeto iCalendar padrão com os valores de propriedades `VERSION:2.0` e `PRODID:-//Unicamp/iScheduler//EN`.

B.2 Algoritmo interativo para Timetable Rearrangement

A biblioteca do framework HuGS, que foi utilizada para implementar o algoritmo interativo para TR, é formada por um pacote principal `hugs` e quatro sub-pacotes:

- `hugs.logging`: contém classes que ajudam a acompanhar o progresso dos algoritmos de busca;
- `hugs.search`: contém as classes que implementam os algoritmos de busca (e.g., `GreedyThread` e `TabuThread`);
- `hugs.support` contém classes de suporte (e.g., `MoveGeneratorMaker`) e de interface de usuário (e.g., `JFramedPanel`); e
- `hugs.util`: contém as classes de utilidades do framework.

As classes e interfaces mais importantes da biblioteca também fazem parte deste pacote principal. As que têm impacto direto na construção de uma aplicação HuGS são:

- `Hugs`: a classe principal do framework. Invoca os algoritmos de busca do pacote `hugs.search`, processa os movimentos manuais do usuário e controla o mecanismo de mobilidades;
- `HugsGUI`: classe que implementa o painel superior da interface, com os objetos gráficos de controle do framework. Os eventos de interface são encaminhados para a classe `Hugs`;

- `Mobilities`: classe que armazena as mobilidades da busca em termos de instâncias da classe `Node`, descrita mais adiante;
- `Move`: é uma interface Java que descreve a execução dos movimentos com dois métodos principais. `getMoved()` retorna um *array* de instâncias da classe `Node` afetadas pelo movimento, e `operateOn(Solution)` executa um movimento em uma solução, alterando-a;
- `MoveGenerator`: é uma interface Java utilizada pelos algoritmos de busca em `hugs.search`. Seus métodos mais importantes são `reset()`, que gera um conjunto de movimentos aplicáveis a uma solução, e `nextMove()`, que retorna o próximo movimento ainda não executado;
- `Node`: classe que corresponde ao conceito de elementos do problema[3]. As mobilidades são associadas a instâncias desta classe, e os movimentos são avaliados com base nos elementos da solução atual que serão alterados;
- `Score`: classe implementa os scores. Seu método principal é `isBetter(Score)`, utilizado para comparar dois scores;
- `Solution`: é uma interface Java que representa a solução de um problema HuGS. O método `getScore()` retorna o score da solução; e
- `Visualization`: é uma classe abstrata herdada de `JFramedPanel` que implementa a camada de visualização do framework. Seu método `setHugs(Hugs)` a conecta à classe central do framework. `setProblem(Problem)` e `setSolution(Solution)` atribuem, respectivamente, o problema e a solução corrente a serem visualizados.

Uma limitação da biblioteca do framework consiste no fato de que são as próprias classes do pacote de busca as responsáveis por atualizar o andamento do algoritmo no painel superior da interface. O fato de estas classes não serem acessíveis diretamente impede, por exemplo, saber exatamente em qual iteração do algoritmo de busca uma nova solução foi encontrada, o que dificulta medições de desempenho das implementações.

O pacote `ischeduler.tr` contém as classes principais que implementam a aplicação HuGS para TR, além dos seguintes sub-pacotes:

- `ischeduler.tr.elements`: contém as classes `Attendee`, que representa os participantes dos eventos, `Event`, que representa os eventos, `PairAttendeeEvent`, que une um participante a um evento para efeito da representação das mobilidades, e `PrecedenceConstraint`, que une dois eventos para representar as relações de precedência $<$ do problema TR;

[3]Talvez um nome mais apropriado para esta classe fosse `Element`.

- `ischeduler.tr.moves`: contém as classes que implementam a interface `Move` do framework. `TRChangeMove` e `TRShiftMove` representam, respectivamente, os movimentos change e shift;

- `ischeduler.tr.nodes`: contém as classes `AttendeeNode` e `EventNode`, que encapsulam os elementos do problema TR aos quais podem ser atribuídos mobilidades. `AttendeeNode` representa um par (`Attendee`, `Event`) associado à mobilidade de um participante em um evento (i.e., a mobilidade de executar um movimento change). `EventNode` representa um evento, associado à sua mobilidade de horário (i.e., a mobilidade de executar um movimento shift); e

- `ischeduler.tr.utils`: contém uma única classe `Utils` de utilidades. Seu método mais importante é `getDefaultProblem()`, que retorna o problema correspondente à instância básica de TR utilizada por Sugihara et al. [77] e Sugumaran et al. [78].

As classes principais de `ischeduler.tr` são:

- `TR`: estende a classe `HuGS`, sobrescrevendo seus métodos principais. O método `makeMoveGenerator` retorna uma classe que gera movimentos específicos do problema TR. O método `makeProblem` retorna a instância do problema TR a ser resolvida, e `makeVisualization` retorna a visualização do problema e de sua solução corrente, a serem mostrados abaixo do painel de controle da interface HuGS;

- `TRMoveGenerator`: implementa a interface `MoveGenerator`, gerando todos os movimentos `TRChangeMove` e `TRShiftMove` que obedecem às mobilidades de uma solução;

- `TRProblem`: implementa a interface `Problem`, representando uma instância do problema TR generalizado, ou seja, os parâmetros P, M_{n+k}, $<'$, t, p e w, e o conjunto $F(\subseteq M_n)$ cujos eventos são inicializados com mobilidade baixa;

- `TRScore`: implementa a interface `Score`, aplicando os critérios de comparação entre duas soluções;

- `TRSolution`: implementa a interface `Solution`, representando uma solução para o problema TR, ou seja, os parâmetros τ e ρ; e

- `TRVisualization`: estende a classe `Visualization`, e utiliza o componente MiG Calendar para implementar a visualização de TR. A classe `Activity` do componente é utilizada para representar as instâncias `Event`, e a classe `Category` modela os `Attendee`s de um `TRProblem`.

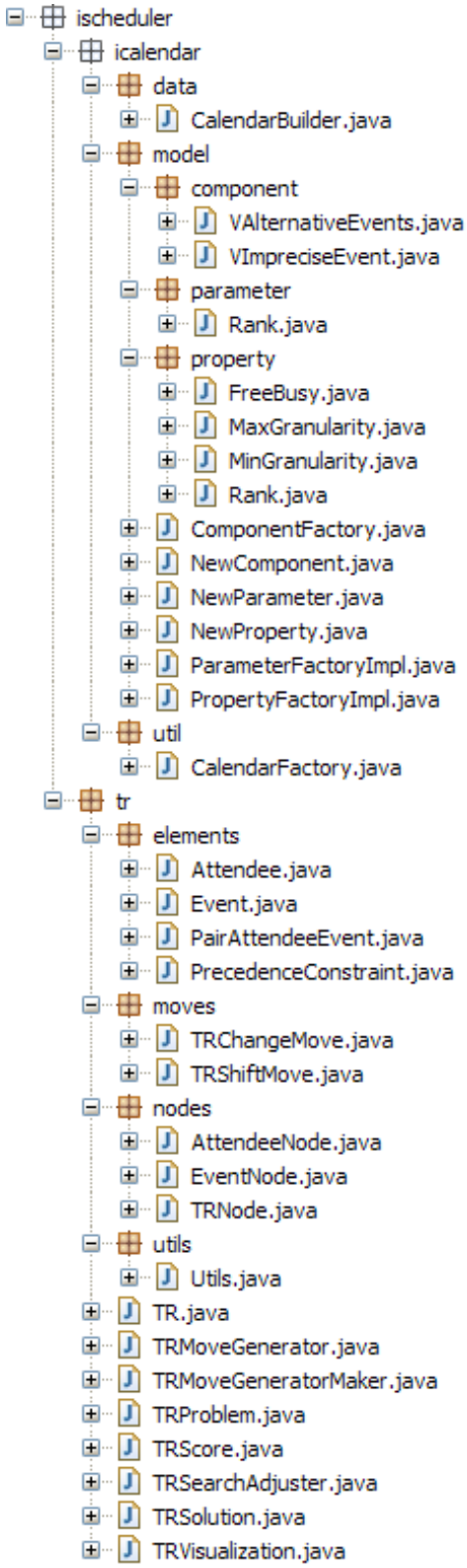


Figura B.1: Hierarquia de classes das implementações realizadas

Bibliografia

[1] Aytug, H., Bhattacharyya, S., Kochlet, G. J., and Snowdon, J. L. A review of machine learning in scheduling. *IEEE Trans. Engineering Management 41*, 2 (May 1994), 165–171.

[2] Baranauskas, M. C. C., and Rocha, H. V. *Design e Avaliação de Interfaces Humano-Computador*. NIED/UNICAMP, 2003.

[3] Barták, R., Müller, T., and Rudová, H. A new approach to modeling and solving minimal perturbation problems. In *Recent Advances in Constraints* (2004), Springer-Verlag, pp. 233–249.

[4] Beard, D., Palaniappan, M., Humm, A., Banks, D., Nair, A., and Shan, Y.-P. A visual calendar for scheduling group meetings. In *Proc. of the Conf. on Computer-Supported Cooperative Work* (Oct. 1990), ACM Press, pp. 279–291.

[5] Berry, P., Peintner, B., Conley, K., Gervasio, M. T., Uribe, T. E., and Yorke-Smith, N. Deploying a personalized time management agent. In *5th International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS)* (Hakodate, Japan, 2006), H. Nakashima, M. P. Wellman, G. Weiss, and P. Stone, Eds., ACM, pp. 1564–1571.

[6] Błażewicz, J., Ecker, K. H., Pesch, E., Schmidt, G., and Weglarz, J. *Scheduling Computers and Manufacturing Processes*. Springer-Verlag, 1996.

[7] Blandford, A., and Green, T. R. G. Group and individual time management tools: What you get is not what you need. *Personal and Ubiquitous Computing 5*, 4 (Dec. 2001), 213–230.

[8] Blum, A. Empirical support for winnow and weighted-majority algorithms: Results on a calendar scheduling domain. *Machine Learning 26*, 1 (1997), 5–23.

[9] Bradner, S. The Internet Standards Process – Revision 3, 1996. RFC 2026.

[10] Brewer, A. C. Interactive scheduling system. *IBM Systems Journal 10*, 1 (1971), 62–79.

[11] CalConnect. Calendar Interoperability Testing Report – Interoperability Testing of RFC 2445, RFC 2446 and RFC 2447, July 2004.

[12] Cooper, T. B., and Kingston, J. H. The complexity of timetable construction problems. In *Selected papers from the First International Conference on Practice and Theory of Automated Timetabling* (London, UK, 1996), Springer-Verlag, pp. 283–295.

[13] Coulouris, G., Dollimore, J., and Kindberg, T. *Distributed Systems: Concepts and Design*. Addison-Wesley, 2001.

[14] Crawford, E., and Veloso, M. Learning dynamic preferences in multi-agent meeting scheduling. In *Proc. of the 2005 IEEE/WIC/ACM Intern. Conf. on Intelligent Agent Technology* (Compiegne, France, Sept. 2005), A. Skowron, J.-P. A. Barthès, L. C. Jain, R. Sun, P. Morizet-Mahoudeaux, J. Liu, and N. Zhong, Eds., IEEE Computer Society, pp. 487–490.

[15] Crawford, S., and Wiers, V. *From Anecdotes to Theory: A Review of Existing Knowledge on Human Factors of Planning and Scheduling*. Taylor & Francis, 2001, pp. 23–57.

[16] Crocker, D., and Overell, P. Augmented BNF for Syntax Specifications: ABNF, 2005. RFC 4234.

[17] Daboo, C. iCalendar Transport-Independent Interoperability Protocol (iTIP). draft-ietf-calsify-2446bis-04, 2007. IETF Internet draft. (Work in progress).

[18] Daboo, C., and Desruisseaux, B. Calendar Availability. draft-daboo-calendar-availability-00, 2007. IETF Internet draft. (Work in progress).

[19] Daboo, C., Desruisseaux, B., and Dusseault, L. CalDAV Scheduling Extensions to WebDAV. draft-desruisseaux-caldav-sched-04, 2007. IETF Internet draft. (Work in progress).

[20] Daboo, C., Desruisseaux, B., and Dusseault, L. Calendaring Extensions to WebDAV (CalDAV), 2007. RFC 4791.

[21] Dawson, F., Mansour, S., and Silverberg, S. iCalendar Message-Based Interoperability Protocol (iMIP), 1998. RFC 2447.

[22] Dawson, F., and Stenerson, D. Internet Calendaring and Scheduling Core Object Specification (iCalendar), 1998. RFC 2445.

[23] Desruisseaux, B. Internet Calendaring and Scheduling Core Object Specification (iCalendar). draft-ietf-calsify-rfc2445bis-07, 2007. IETF Internet draft. (Work in progress).

[24] do Nascimento, H. A. D., and Eades, P. User hints: a framework for interactive optimization. *Future Generation Comp. Syst. 21*, 7 (July 2005), 1171–1191.

[25] Dusseault, L. HTTP Extensions for Web Distributed Authoring and Versioning (WebDAV), 2007. RFC 4918.

[26] Dusseault, L., and Whitehead, J. Open Calendar Sharing and Scheduling with CalDAV. *IEEE Internet Comput. 9*, 2 (Mar. 2005), 81–89.

[27] Egen, P. An Introduction to Calendaring Standards - How it all started, 2005. CalConnect Consortium.

[28] Egen, P. Calendar Interoperability Testing Report - Public - September 2007 - Boston, MA - Sponsored by MIT, 2007. CalConnect Consortium.

[29] Ehrlich, S. F. Strategies for encouraging successful adoption of office communication systems. *ACM Trans. on Office Inf. Syst 5*, 4 (Oct. 1987), 340–357.

[30] Erickson, T. From PIM to GIM: personal information management in group contexts. *Commun. ACM 49*, 1 (Jan. 2006), 74–75.

[31] Fenner, B. Guidelines to Authors of Internet-Drafts, 2006. http://www.ietf.org/ietf/1id-guidelines.html.

[32] Fielding, R. Hypertext Transfer Protocol - HTTP/1.1, 1999. RFC 2616.

[33] Garey, M. R., and Johnson, D. S. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman and Company, 1979.

[34] Goltz, H., and Matzke, D. Combined interactive and automatic timetabling. In *Proc. PACLP99* (1999), pp. 529–535.

[35] Greif, I., and Sarin, S. K. Data sharing in group work. *ACM Trans. on Office Inf. Sys. 5*, 2 (Apr. 1987), 187–211.

[36] Grimes, J. E. Scheduling to reduce conflict in meetings. *Commun. ACM 13*, 6 (June 1970), 351–352.

[37] Grudin, J. Why CSCW applications fail: problems in the design and evaluation of organization of organizational interfaces. In *CSCW '88: Proceedings of the 1988 ACM conference on computer-supported cooperative work* (New York, USA, 1988), ACM Press, pp. 85–93.

[38] Higgins, P. G. Human-computer production scheduling: Contribution to the hybrid automation paradigm. In *Proc. First. Int. Conf. Ergonomics of Advanced Manufacturing and Hybrid Automated Systems* (1992), Elsevier, pp. 211–216.

[39] Horvitz, E. Principles of mixed-initiative user interfaces. In *Proc. ACM SIGCHI Conf. Human Factors in Computing Systems* (Pittsburgh, PA, 1999), ACM Press, pp. 159–166.

[40] Internet Mail Consortium. vCalendar - The Electronic Calendaring and Scheduling Exchange Format, 1996.

[41] Kelley, J., and Chapanis, A. How professional persons keep their calendars: Implications for computerization. *Journal of Occupational Psychology 55* (1982), 241–256.

[42] Kincaid, C. M., Dupont, P. B., and Kaye, A. R. Electronic calendars in the office: an assessment of user needs and current technology. *ACM Trans. Inf. Syst. 3*, 1 (Jan. 1985), 89–102.

[43] Klau, G. W., Lesh, N., Marks, J., and Mitzenmacher, M. Human-guided tabu search. In *Proc. 18th National Conf. Artificial Intelligence, AAAI* (Menlo Parc, CA, USA, July 2002), AAAI Press, pp. 41–47.

[44] Klau, G. W., Lesh, N., Marks, J., Mitzenmacher, M., and Schafer, G. T. The HuGS platform: A toolkit for interactive optimization. Tech. Rep. TR2002-08, Mitsubishi Electric Research Laboratories, June 2002.

[45] Kozierok, R., and Maes, P. A learning interface agent for scheduling meetings. In *Proceedings of the International Workshop on Intelligent User Interfaces* (New York, USA, Jan. 1992), W. D. Gray, W. E. Hefley, and D. Murray, Eds., ACM Press, pp. 81–88.

[46] Lamport, L. Time, clocks, and the ordering of events in a distributed system. *Commun. ACM 21*, 7 (1978), 558–565.

[47] Lee, C.-S., and Pan, C.-Y. An intelligent fuzzy agent for meeting scheduling decision support system. *Fuzzy Sets and Systems 142* (2004), 467–488.

[48] Lesh, N., Lopes, L. B., Marks, J., Mitzenmacher, M., and Schafer, G. T. Human-guided search for jobshop scheduling. Tech. Rep. TR2002-43, Mitsubishi Electric Research Laboratories, 2003.

[49] Mahoney, B., Babics, G., and Taler, A. Guide to Internet Calendaring, 2002. RFC 3283.

[50] McCullough, J. Calendaring and scheduling glossary of terms, 2006. CalConnect Consortium.

[51] Melnikov, A. iCalendar Message-Based Interoperability Protocol (iMIP). draft-ietf-calsify-rfc2447bis-03, 2007. IETF Internet draft. (Work in progress).

[52] Mitchell, T., Caruana, R., Freitag, D., McDermott, J., and Zabowski, D. Experience with a learning personal assistant. *Commun. ACM 37*, 7 (July 1994), 81–91.

[53] Modi, P., and Veloso, M. Multiagent meeting scheduling with rescheduling. In *The 5th International Workshop on Distributed Constraint Reasoning* (Toronto, Canada, 2004), pp. 198–201.

[54] Modi, P. J., Veloso, M. M., Smith, S. F., and Oh, J. CMRadar: A personal assistant agent for calendar management. In *AOIS* (2004), P. Bresciani, P. Giorgini, B. Henderson-Sellers, G. Low, and M. Winikoff, Eds., vol. 3508 of *Lecture Notes in Computer Science*, Springer, pp. 169–181.

[55] Mosier, J. N., and Tammaro, S. G. When are group scheduling tools useful? *Computer Supported Cooperative Work 6*, 1 (Mar. 1997), 53–70.

[56] Mueller, E. T. A calendar with common sense. In *IUI '00: Proceedings of the 5th international conference on intelligent user interfaces* (New York, NY, USA, 2000), ACM Press, pp. 198–201.

[57] Netscape. More than 20 companies join Netscape to help define open standard for Internet calendaring and scheduling, 1996. Netscape Communications. (Press release).

[58] Netscape. Overview of calendaring and scheduling standards, 2000. Netscape White Papers.

[59] Norris, C., and McCullough, J. Internet calendaring and scheduling venue component specification. draft-norris-ical-venue-01, 2007. IETF Internet draft. (Work in progress).

[60] Oh, J., and Smith, S. Learning user preferences in distributed calendar scheduling. In *Proc. 5th International Conference on Practice and Theory of Automated Timetabling (PATAT)* (Pittsburgh, PA, USA, 2004), Springer-Verlag, pp. 35–50.

[61] Palen, L. Social, individual and technological issues for groupware calendar systems. In *CHI* (Pittsburgh, PA, USA, 1999), ACM Press, pp. 17–24.

[62] Pinedo, M. *Scheduling: theory, algorithms, and systems.* Prentice-Hall, 2002.

[63] Pinedo, M., and Chao, X. *Operations Scheduling with Applications in Manufacturing and Services*. Irwin/McGraw-Hill, 1999.

[64] Reynolds, J., and Braden, R. Instructions to Request for Comments (RFC) Authors. draft-rfc-editor-rfc2223bis-08, 2004. IETF Internet draft. (Work in progress).

[65] Royer, D., Babics, G., and Mansour, S. Calendar access protocol (CAP), 2005. RFC 4324.

[66] Russell, S., and Norvig, P. *Artificial Intelligence: a Modern Approach*. Prentice-Hall, 2002.

[67] Sayers, C., and Letsinger, R. The CoolAgent ontology: A language for publishing and scheduling events. Tech. Rep. HPL-2001-194, Hewlett Packard Laboratories, 2001.

[68] Sayers, C., and Letsinger, R. An ontology for publishing and scheduling events and the lessons learned in developing it. Tech. Rep. HPL-2002-162, Hewlett Packard Laboratories, 2002.

[69] Schultz, C. K., Brooks, A., and Schwartz, P. Scheduling meetings with a computer. *Commun. ACM 7*, 9 (Sept. 1964), 534–541.

[70] Sen, S. Developing an automated distributed meeting scheduler. *IEEE Expert 12*, 4 (1997), 41–45.

[71] Sen, S., and Durfee, E. H. A formal study of distributed meeting scheduling. *Group Decision and Negotiation 7*, 3 (May 1998), 265–289.

[72] Shintani, T., Ito, T., and Sycara, K. P. Multiple negotiations among agents for a distributed meeting scheduler. In *ICMAS* (2000), IEEE Computer Society, pp. 435–436.

[73] Silva, F. Framework gráfico para um escalonador genérico de recursos, 2004. Relatório da disciplina MC030 - Projeto Final de Graduação, Unicamp.

[74] Silva, F., and Drummond, R. Imprecise and alternative events for iCalendar and iTIP. draft-silva-events-01, 2007. IETF Internet draft. (Work in progress).

[75] Silva, F., and Drummond, R. New approaches for groupware scheduling. Tech. Rep. IC-07-20, Instituto de Computação - Unicamp, June 2007.

[76] Silverberg, S., Mansour, S., Dawson, F., and Hopson, R. iCalendar Transport-Independent Interoperability Protocol (iTIP) – Scheduling Events, BusyTime, To-dos and Journal Entries, 1998. RFC 2446.

[77] Sugihara, K., Kikuno, T., and Yoshida, N. A meeting scheduler for office automation. *IEEE Trans. Softw. Eng. 15*, 10 (Oct. 1989), 1141–1146.

[78] Sugumaran, M., Easawarakumar, K. S., and Narayanasamy, P. A New Approach for Meeting Scheduler using A*-Algorithm. In *TENCON 2003. Conference on Convergent Technologies for Asia-Pacific Region* (2003), vol. 1, pp. 419–423.

[79] Sugumaran, M., Easawarakumar, K. S., and Narayanasamy, P. A Heuristic Approach for Rescheduling of Meetings with A-Algorithm. *Asian Journal of Information Technology 5*, 6 (2006), 640–646.

[80] Sugumaran, M., Easawarakumar, K. S., and Narayanasamy, P. An Effective Approach for Distributed Meeting Scheduler. *International Journal of Information Technology 12*, 8 (2006), 73–92.

[81] Swartz, L. Why people hate the paperclip: Labels, appearance, behavior, and social responses to user interface agents, 2003. Honors Thesis for Symbolic Systems Program, Stanford University.

[82] Tanenbaum, A. S. *Computer Networks*, 4th ed. Prentice-Hall, 2003.

[83] Vazirani, V. V. *Approximation Algorithms*. Springer-Verlag, 2003.

[84] Wijnen, B. Checklist for Internet-Drafts (IDs) submitted for RFC publication, 2006. http://www.ietf.org/ID-Checklist.html.

[85] Wren, A. Scheduling, timetabling and rostering - a special relationship? In *Proceedings of the First International Conference on the Practice and Theory of Automated Timetabling (ICPTAT '95)* (1995), pp. 474–495.

[86] Zweben, M., and Fox, M. S. *Intelligent Scheduling*. Morgan Kaufmann, 1994.