Groupware: Conceitos Fundamentais e Caracterização dos Principais Blocos Construtivos

Pedro Antunes

DI-FCUL TR–02–16

Novembro 2002

Departamento de Informática
Faculdade de Ciências da Universidade de Lisboa
Campo Grande, 1700 Lisboa
Portugal

Technical reports are available at http://www.di.fc.ul.pt/tech-reports. The files are stored in PDF, with the report number as filename. Alternatively, reports are available by post from the above address.

Groupware: Conceitos Fundamentais e Caracterização dos Principais Blocos Construtivos

Pedro Antunes

Departmento de Informática
Faculdade de Ciências da Universidade de Lisboa

Bloco C5 – Piso 1 – Campo Grande, 1700 Lisboa, Portugal

paa@di.fc.ul.pt, www.di.fc.ul.pt/~paa

Resumo

Este trabalho parte de um conjunto de conceitos fundamentais da área de CSCW (*Computer Supported Cooperative Work*) e procura identificar os blocos construtivos requeridos por qualquer sistema, ferramenta ou aplicação informática que se queira considerar cooperativa, à qual genericamente se designa groupware. Os blocos considerados são: comunicação, arquitectura, concorrência, coordenação, espaço público, monitorização e acesso.

Índice

1 Introdução

A disseminação dos sistemas computacionais pelos ambientes organizacionais tem apresentado uma tendência crescente ao longo dos anos. Esta tendência foi possibilitada pela banalização, em primeiro lugar, do equipamento, e, em segundo lugar, das infraestruturas de comunicações que permitem interligar postos de trabalho, grupos de trabalho, departamentos de organizações e organizações.

Este ambiente organizacional, povoado de computadores autónomos mas interligados, instigou ao surgimento de sistemas especificamente voltados para o trabalho em grupo, vulgarmente designados por groupware (por analogia a hardware e software).

Por exemplo, a utilização de sistemas de correio electrónico, sistemas de boletins, audio e video conferência, sistemas de calendarização de tarefas em grupo e sistemas de fluxos de trabalho (*workflow*) está a tornar-se comuns nas actividades diárias das organizações.

Ora, a actividade em grupo resulta de processos sociais complexos de decisão, negociação, resolução de conflitos, planeamento e desenvolvimento de actividades e aprendizagem. Todos estes processos sociais requerem um mecanismo fundamental a que chamamos interacção: partilha de ideias, valores, sentidos e estabelecimento de uma compreensão mútua entre os diversos participantes nos processos (Patton et al., 1989).

A interacção, sendo fundamentalmente necessária ao desenrolar das actividades nas organizações, é também um factor importante nas relações sociais e humanas. Consequentemente, o desenho de groupware deve considerar a complexidade de factores e requisitos sociais associados à interacção entre os indivíduos, grupos e organizações. Caso contrário, a tecnologia falha por falta de motivação no seu uso.

Por exemplo, quando os indivíduos interactuam face-a-face, são utilizados múltiplos canais de comunicação: audíveis, visuais, faciais, movimentos corporais ou sinais psicológicos. As mensagens trocadas através destes canais são compreendidas por quem as envia e recebe, uma tarefa que implica percepção, cognição e julgamento de cada indivíduo. Quando os indivíduos interactuam através de groupware encontram diversas restrições à diversidade de símbolos que podem ser trocados e à disponibilidade e largura de banda dos canais utilizados. Essas restrições podem ser suficientes para aumentar a conflitualidade, equívocos, falta de persuasão, incapacidade negocial e, consequentemente, levar ao abandono do sistema.

Outra questão importante é a interdependência entre os elementos de um grupo. As actividades em grupo são interdependentes no sentido em que fluem reciprocamente de um indivíduo para outro (Butler, 1991). Mas, ao contrário da interacção face-a-face, onde a interdependência é gerida de forma implícita pelos participantes, o groupware torna esta gestão explícita ao introduzir mecanismos de coordenação. A questão está pois em identificar onde deve estar o controlo, se nos utilizadores ou no sistema, sabendo-se que demasiado controlo do sistema pode resultar em sistemas com demasiadas excepções e consequentes abandonos.

A cooperatividade é a tendência dos membros de um grupo para cooperar na resolução de um problema. Os conflitos entre membros de um grupo devem ser evitados pois, quando em excesso, podem restringir o seu desempenho, levando à desconfiança e insegurança. Pelo contrário, a cooperatividade leva à coordenação de esforços, produtividade, boas relações humanas e outros efeitos positivos (Patton et al., 1989).

Existem provas de que o groupware pode limitar a cooperatividade dos participantes (Grudin, 1994). Para ultrapassar este problema não só é necessário promover o uso da tecnologia, através de serviços que de outro modo não estariam disponíveis aos utilizadores (automatizar algumas tarefas, por exemplo), como também é necessário que esses serviços não beneficiem uns em detrimento de outros.

Como se pode observar pelos exemplos dados, quem desenvolve groupware depara-se com um conjunto muito vasto e diversificado de problemas. A área de investigação que se preocupa com a identificação, análise e resolução deste tipo de problemas é a área de CSCW (*Computer Supported Cooperative Work*).

1.1 Perspectiva histórica

Considera-se que o primeiro sistema a abordar explicitamente a cooperação foi o sistema NLS/Augment, demonstrado em 1962 no Stanford Research Institute e desenvolvido sobre a responsabilidade de Douglas Engelbart (Engelbart e English, 1988).

Interessado desde 1952 no tema da extensão da capacidade dos indivíduos pela via da tecnologia, Douglas Engelbart desenvolveu um cenário conceptual que caracterizou deste modo: “I visualized people collaborating interactively on visual displays connected to a computer complex. I’m not ‘numerically oriented’, my vision has always facilitated discursive thinking and collaboration.”

A designação CSCW foi proposta por Irene Greif e Paul Cashman em 1984 (Bannon, 1993), tendo em vista descrever um sistema ou o desenvolvimento de sistemas que suportassem as pessoas no desenrolar das suas actividades.

O tema atraiu desde o início investigadores de áreas muito diversas dentro da informática, como sejam sistemas de informação organizacionais, sistemas distribuídos, hipertexto e hipermédia, interacção humano-computador, comunicação mediada por computador ou inteligência artificial. A esses, devem-se ainda junto investigadores de outras áreas, como sejam gestão organizacional, sociologia, etnologia, psicologia. O interesse destes investigadores pela área de CSCW deve-se concerteza à combinação muito particular de tecnologia, pessoas e organizações que a área engloba.

Sem prejuízo de outros, pode-se considerar que os vectores actualmente mais influentes na área de CSCW são (Bannon, 1993):

- Os factores sociais e humanos associados ao uso de sistemas cooperativos, incluindo a produtividade, usabilidade e utilidade destes sistemas.
- Os factores tecnológicos, englobando o desenvolvimento de tecnologia inovadora de suporte a interacção, coordenação, partilha de informação e interface pessoa-máquina.
- A mudança de paradigma no desenvolvimento de sistemas, em particular recentrando o desenvolvimento nos utilizadores, na identificação das suas práticas e do contexto onde realizam trabalho.

1.2 Caracterização genérica

A questão que iremos aqui abordar é a de caracterizar genericamente a tecnologia de groupware. Uma opinião que tem prevalecido na literatura é que esse papel é difícil de definir devido à diversidade de domínios e aplicações deste tipo de tecnologia, que cobrem àreas tão diversas como o suporte à comunicação, coordenação, decisão, negociação, planeamento, gestão, desenvolvimento de software, pesquisa cooperativa de informação, etc. A caracterização funcional da tecnologia de suporte a cooperação torna-se assim extremamente complexa.

Seguramente que a categorização que mais impacto tem demonstrado é a que recorre às dimensões espaço/tempo (Ellis et al., 1991; Johansen et al., 1991). A combinação das dimensões de tempo e espaço estabelece quatro tipos de groupware com características bem distintas (Figura 1.1). Os sistemas que se posicionam no quadrante tempo-igual/local-igual caracterizam-se pela interacção face-a-face entre os utilizadores extendida

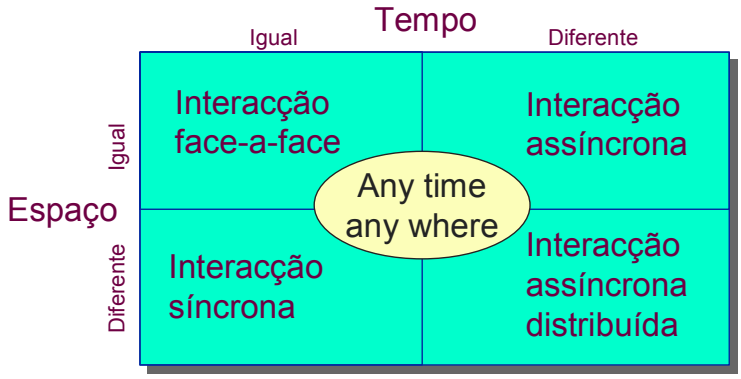


Figura 1.1: Classificação espaço/tempo

pelo suporte computacional à geração, partilha e manipulação de informação em grupo. O quadro electrónico (*whiteboard*; Elrod et al., 1992) é um exemplo deste tipo de groupware.

Os sistemas que se posicionam no quadrante tempo-diferente/local-diferente caracterizam-se pela interacção assíncrona entre os utilizadores. Nesta situação as actividades são realizadas individualmente, servindo o sistema de mediador, permitindo a troca de informação e a coordenação de actividades. O correio electrónico e os sistemas de boletins constituem exemplos deste tipo de sistemas.

Os sistemas de tipo tempo-diferente/local-igual, dada a sua natureza própria, restringem consideravelmente a interacção entre os seus utilizadores. Neste quadrante podem por exemplo posicionar-se quiosques de cooperação (Johansen et al., 1991) ou blocos de notas partilhados por equipas que rodam no posto de trabalho sem se encontrarem (Robinson et al., 1998).

Os sistemas de tipo tempo-igual/local-diferente, devido a suportarem a interacção síncrona entre utilizadores remotos do sistema, são aqueles que mais exigem do ponto de vista de suporte computacional. A comunicação e partilha de informação são essenciais para manter o contexto partilhado entre os utilizadores; o sistema é também essencial para coordenar as actividades dos utilizadores.

Finalmente, devem ainda considerar-se a existência de sistemas suficientemente flexíveis para permitirem uma utilização em qualquer uma das situações de espaço e tempo referenciadas. Estes sistemas são designados por *any-time/any-where*.

Uma outra classificação da tecnologia de groupware foi proposta por Nunamaker et al. (1997). Em primeiro lugar, procura-se analisar o tipo de ligação entre os membros de um grupo, de que resulta uma classificação do trabalho em grupo em três níveis distintos:

- Nível individual – Nesta situação de trabalho em grupo o esforço realiza-se apenas a nível individual, não havendo necessidade de ser coordenado para atingir um objectivo. É o caso de uma representação nacional nas provas de atletismo em que todos os elementos nas respectivas classes fazem esforço no sentido de obterem medalhas. O resultado final é portanto a soma dos resultados individuais.
- Nível coordenado – Neste nível de trabalho em grupo o esforço é individual mas coordenado. É o que acontece nas corridas de estafetas, em que os corredores correm individualmente mas devem estar coordenados no momento de troca de testemunho.
- Nível concertado – Neste nível o esforço é realizado de forma conjunta para atingir uma meta. É o que acontece com os remadores numa embarcação.

Por outro lado, podem também classificar-se as diferentes formas como os membros de um grupo realizam as suas tarefas:

- Comunicação – A comunicação envolve a escolha de um conjunto de palavras, comportamentos e imagens e a sua transmissão através de um meio adequado para que a informação seja recebida e entendida pelos elementos do grupo.
- Reflexão – A reflexão é um processo longo e complexo que, numa perspectiva racionalista, envolve a realização de uma série de tarefas que cobrem desde a definição de intenções até à realização de metas.
- Acesso a informação – O acesso a informação envolve a procura de informação que os membros do grupo necessitam para apoiar a sua reflexão.

Combinando as duas classificações apresentadas anteriormente, surge então a matriz do Arizona (Nunamaker et al., 1997). Esta matriz pode assim ser utilizada para classificar diversas tecnologias de groupware (Figura 1.2).

Trabalho em grupo				
	Dinâmica de grupo	Sistemas de comunicação de audio e vídeo em redes dedicadas, incluindo audio e video conferência; Comunicação multimédia, suportada por redes de computadores; Sistemas de conversação persistente (*chat*).	Sistemas de suporte a reuniões electrónicas e sistemas de suporte à decisão em grupo, incluindo ferramentas de brainstorming, definição de critérios e votação.	Quadros electrónicos partilhados (*shared whiteboards*); Editores multimédia, incluindo editores de hipertexto; Sistemas que organizam espacialmente as actividades dos utilizadores, incluindo os sistemas MOO (Multi-User Domain Object-Oriented).
	Coordenação	Sistemas de gestão de correio electrónico, incluindo sistemas de boletins e fora de discussão.	Calendários de grupo e sistemas para marcação de reuniões; Sistemas de coordenação de tarefas; Sistemas de fluxos de trabalho (*workflow*).	Sistemas de partilha de artefactos diversos, como documentos e recomendações.
	Individual	Sistemas de localização de pessoas.	Modelação; Simulação.	Filtragem de informação.
		Comunicação	Reflexão	Acesso a Informação
		tarefas		

Figura 1.2: Matriz do Arizo

1.3 Abordagem

Neste documento procurarei identificar alguns blocos construtivos que considero fundamentais em qualquer sistema que se queira considerar cooperativo. Os blocos considerados são:

- Comunicação – A comunicação de dados é um dos elementos essenciais de um sistema cooperativo (exceptuando talvez os sistemas mesmo-local/tempo-diferente), sem o qual não é possível suportar a interacção entre os utilizadores do sistema. Importa portanto identificar que características relacionadas com a comunicação de dados determinam o desempenho dos sistemas cooperativos.
- Arquitectura – A arquitectura de um sistema cooperativo influencia significativamente a funcionalidade de um sistema cooperativo, isto porque condiciona todo o mecanismo de comunicação de dados no sistema. Importa pois identificar como se pode alinhar a arquitectura com a funcionalidade pretendida para o sistema cooperativo.
- Concorrência – Sendo a informação manipulada por múltiplos utilizadores, levanta-se naturalmente o problema de preservar a sua coerência. Os dados partilhados devem ser geridos de forma a optimizar a manipulação concorrente, permitindo actividades paralelas mas, simultaneamente, possibilitando que os utilizadores organizem as suas actividades.

- Coordenação – O suporte à coordenação entre os utilizadores do sistema é fundamental para garantir a interdependência. Importa pois identificar que mecanismos de coordenação se encontram disponíveis.
- Espaços públicos – Os espaços públicos, partilhados por todos os utilizadores, devem ser geridos de forma a que a coerência (visual) dos objectos neles contidos seja mantida. Este problema é semelhante ao do controlo da concorrência, sendo todavia necessário cuidar de alguns requisitos particulares da interacção pessoa-máquina, nomeadamente, o tempo de resposta do sistema aos utilizadores.
- Monitorização – A interface pessoa-máquina é não só responsável por mediar a relação entre o utilizador e o sistema como também por actuar como mediadora entre os utilizadores do sistema. Torna-se assim fundamental que esta permita uma adequada monitorização das acções realizadas pelos utilizadores.
- Acesso – Os mecanismos de acesso permitem ao sistema cooperativo planear e gerir, a alto nível, os papeis e actividades dos utilizadores do sistema.

2 Comunicação

Do ponto de vista dos requisitos de comunicação de dados, um sistema cooperativo não difere substancialmente de um sistema distribuído, pelo que este aspecto não será esmiuçado. Utilizaremos o modelo de comunicação típico que considera a existência de quatro elementos fundamentais na comunicação: emissor, receptor, mensagem e canal.

Os sistemas cooperativos utilizam três tipos distintos de comunicação (Ochoa et al., 2002):

- Ponto-a-ponto – Comunicação entre dois nós do sistema.
- Multiponto (*multicast*) – Comunicação entre diversos nós do sistema.
- Difusão (*broadcast*) - Comunicação entre todos os nós do sistema.

Para além disso, devem ser considerados dois modos de entrega de mensagens: síncrono e assíncrono. No caso do modo síncrono considera-se que as mensagens são transmitidas do emissor para o receptor sem que elas sejam explicitamente guardadas pelo canal de comunicação. Um caso especial do modo síncrono ocorre quando as mensagens são transmitidas e recebidas em intervalos regulares (*streaming* de audio e video). No caso do modo assíncrono considera-se que as mensagens são explicitamente guardadas pelo canal.

Ums dos aspectos relacionados com comunicação e que é fundamental para um sistema cooperativo é o desempenho do canal na entrega de mensagens. Aqui temos de considerar diversos critérios para medir esse desempenho:

- Tempo de notificação – O tempo que decorre desde que uma mensagem é enviada por um nó até que essa mensagem é entregue a outro nó (*feedthrough* da mensagem). O tempo de notificação determina o grau de interactividade entre os utilizadores de um sistema cooperativo.
- Tempo de resposta – O tempo que decorre desde que uma mensagem é enviada por um nó até que uma resposta é entregue a esse mesmo nó (*feedback* da mensagem). O tempo de resposta determina a capacidade do canal para suportar comunicação bi-direccional entre emissores e receptores. Num âmbito mais alargado, o tempo de resposta afecta a flexibilidade dos utilizadores do sistema em

interromperem, alterarem ou terminarem a comunicação (Dennis et al., 1998).

- Largura de banda (*throughput*) – Número de mensagens que um nó consegue enviar num período especificado de tempo. A largura de banda determina a granularidade dos eventos do sistema (número de eventos relevantes que um nó pode transmitir).
- Endereçamento múltiplo - Número de nós que podem estabelecer ligação ao canal, o que condiciona a escalabilidade do sistema (Dennis et al., 1998).

Note-se que os canais de comunicação podem ser independentes ou inter-dependentes. Está demonstrado que alguns sistemas cooperativos necessitam de canais inter-dependentes, de forma a garantir a sua sincronização (Teixeira et al., 2002).

3 Arquitectura

A partilha de informação é um dos objectivos fundamentais de um sistema cooperativo. Iremos aqui identificar e analisar as diferentes arquitecturas que permitem atingir esse objectivo.

Do ponto de vista da arquitectura do sistema, entende-se como partilha de informação a capacidade de disseminar dados por múltiplos nós que se encontram interligados por uma rede de comunicação de dados. Cada nó é tipicamente constituído por uma estação de trabalho manipulada por um único utilizador.

Importa então identificar as possíveis alternativas para organizar a informação no conjunto de nós do sistema (Sarin e Greif, 1985; Greenberg, 1990; Lauwers e Lantz, 1990; Lauwers et al., 1990).

3.1 Arquitectura centralizada mono-utilizador

A ideia original associada a este tipo de arquitectura foi partilhar uma qualquer aplicação mono-utilizador por diversos utilizadores sem, no entanto, proceder a qualquer modificação ou extensão dessa aplicação. Como a Figura 3.1 ilustra, o ambiente onde a aplicação (no caso, uma

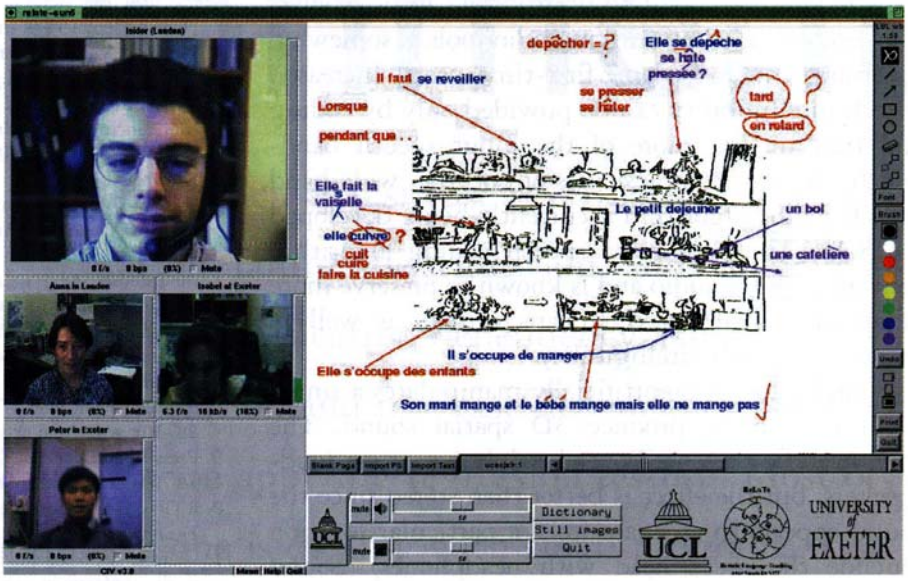


Figura 3.1: Ambiente típico de partilha de uma aplicação mono-utilizador

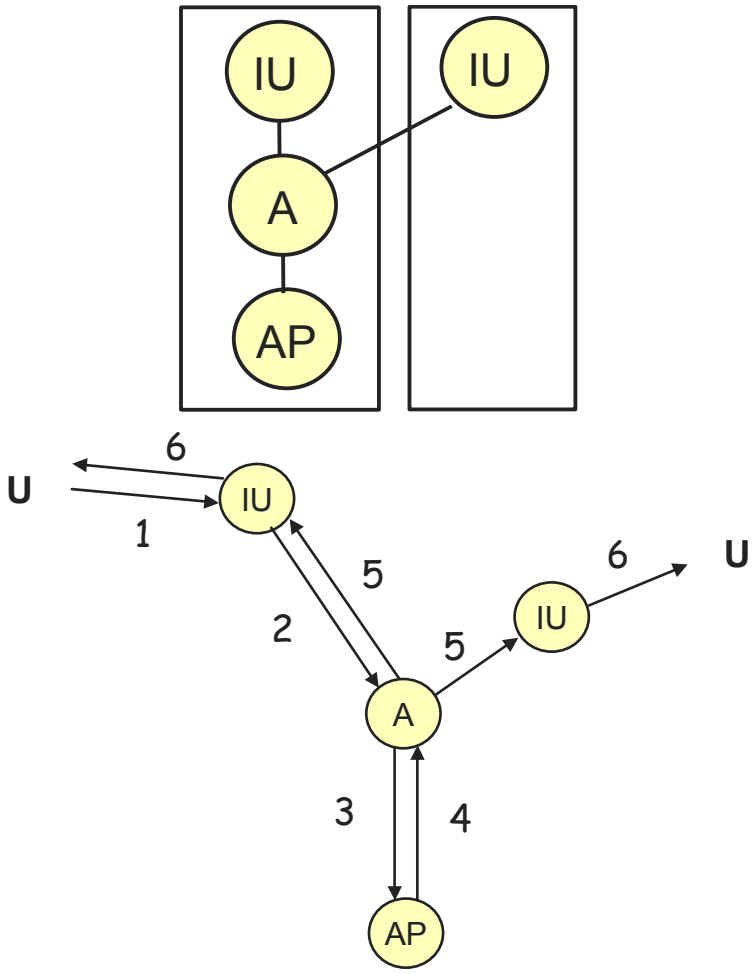


Figura 3.2: Partilha de informação na arquitectura centralizada mono-utilizador

aplicação de desenho) é partilhada tem que fornecer canais de comunicação adicionais, por exemplo video-conferência. Esta necessidade decorre das limitações próprias da arquitectura centralizada mono-utilizador: como a aplicação não suporta a colaboração entre os utilizadores no acesso aos seus dados e funcionalidades, terá que ser o ambiente a suprir essa falta.

A um nível abstracto, pode-se conceber a arquitectura do sistema com sendo constituída por (Figura 3.2) um agente (A) que se coloca entre a aplicação (AP) e a camada de interface com o utilizador (IU).

A função do agente é multiplexar as saídas da aplicação. O agente intercepta as saídas da aplicação destinadas à interface com o utilizador (4). A intercepção é realizada ao nível da biblioteca do sistema de janelas, sendo por exemplo facilmente realizável no X-Windows. Em seguida, o agente

dissemina as saídas pelos múltiplos nós do sistema de forma transparente para a aplicação (5).

Note-se que, como a aplicação não sabe que existem múltiplos utilizadores, não é possível personalizar a informação que lhes é mostrada (Bentley et al., 1992, 1994). Esta é a primeira restrição fundamental imposta pela arquitectura centralizada mono-utilizador.

No que se refere às entradas (rato, teclado, etc), estas são igualmente interceptadas pelo agente que se encarrega de as enviar para a aplicação (1, 2, 3). É normal que, tendo em vista a manutenção de alguma coerência neste processo de partilha, o agente mantenha apenas um canal de entrada activo. Caberá ao ambiente de partilha definir que utilizador deve estar activo, ficando os outros utilizadores em situação passiva a visualizar a aplicação, um tipo de controlo designado por controlo de palco. Esta é a segunda restrição imposta por esta arquitectura.

Finalmente, a terceira restrição a considerar deriva do tempo de resposta do sistema (por exemplo, desde que o utilizador pressiona um botão até que este muda de côr). Numa situação em que IU, A e AP se encontram geograficamente distribuídos, com um tempo de resposta elevado, os utilizadores ficam limitados na sua capacidade para interagirem com o sistema e, eventualmente, rejeitam-no.

3.2 Arquitectura centralizada multi-utilizador

As diferenças deste tipo de arquitectura em relação à arquitectura

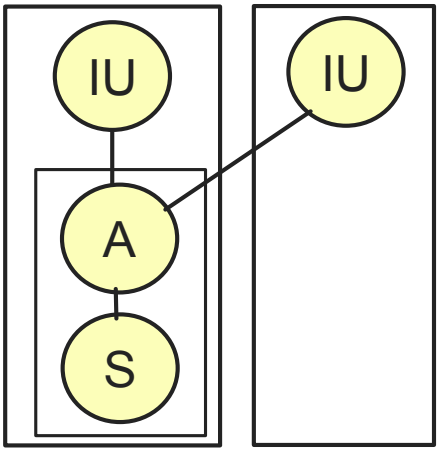


Figura 3.3: Arquitectura centralizada multi-utilizador

centralizada mono-utilizador é que agora a aplicação reconhece explicitamente que existem múltiplos utilizadores. A aplicação pode então ser decomposta em duas camadas bem distintas (Figura 3.3), uma correspondente ao agente (A), dedicada a disseminar a informação pelos nós cliente, e a outra correspondente à semântica da aplicação (S).

Comparativamente à arquitectura anterior, esta solução permite suportar actividades concorrentes dos utilizadores, personalizar a informação que lhes é dirigida, assim como gerir o acesso dos utilizadores tendo por base a semântica da própria aplicação. Porém, o problema do tempo de resposta mantém-se.

3.3 Arquitectura distribuída

Na arquitectura distribuída a aplicação encontra-se distribuída pelos nós do sistema, podendo-se mais uma vez distinguir as camadas correspondentes

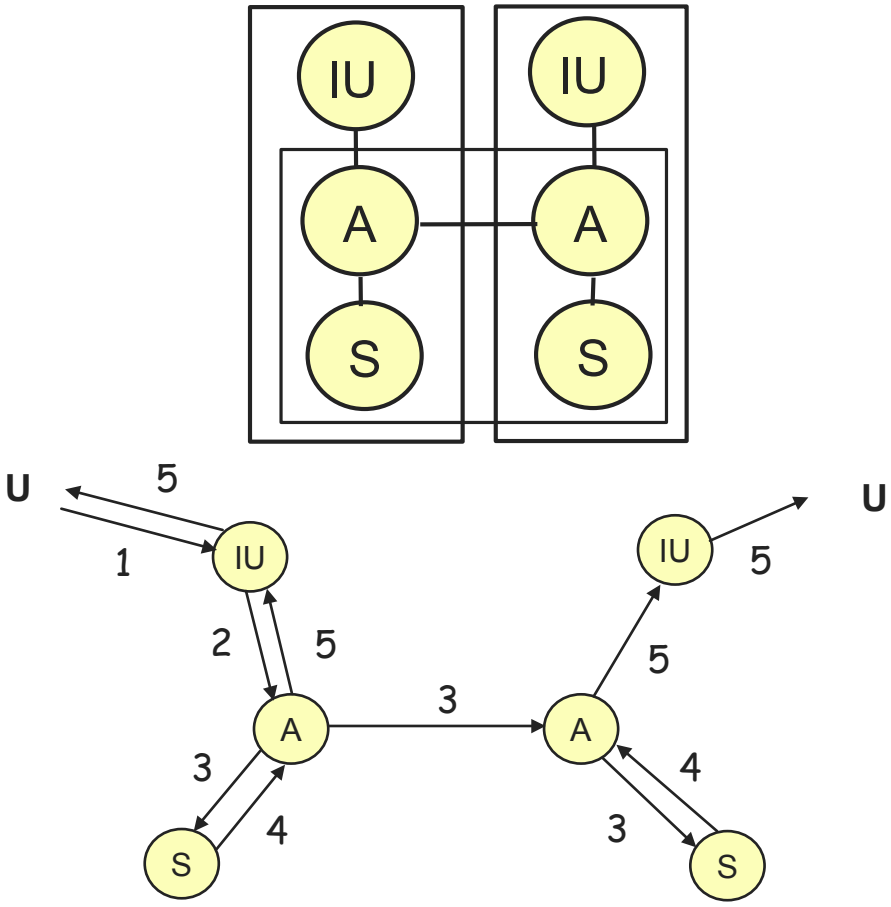


Figura 3.4: Arquitectura distribuída

aos agentes e semântica da aplicação (Figura 3.4).

Neste tipo de arquitectura as entradas dos utilizadores (1, 2) são enviadas directamente a todos os nós pelo agente local (3), o que, comparativamente à arquitectura centralizada, elimina uma indirecção na transmissão de dados.

Acresce ainda que as saídas da aplicação para os utilizadores podem ser geradas localmente a cada nó, o que resolve o problema do tempo de resposta (4, 5).

Um outro aspecto importante a considerar é a granularidade da informação trocada entre nós. Se, no caso da arquitectura centralizada mono-utilizador, as mensagens eram trocadas ao nível do sistema de janelas, agora as mensagens trocadas entre os diversos nós podem ser de alto nível, o que pode reduzir consideravelmente o tráfego de mensagens na rede.

Testes realizados com uma aplicação específica revelaram que a versão centralizada gerava 3.6 vezes mais mensagens na rede de dados do que a versão distribuída da mesma aplicação (Ahuja et al., 1990). Esta desproporção elevou-se para 6 quando aumentou o peso da informação gráfica na aplicação.

Outras vantagens consideráveis da arquitectura distribuída são uma maior escalabilidade, disponibilidade e fiabilidade do sistema. Por exemplo, é natural que sistemas de elevada escalabilidade utilizem mais do que um

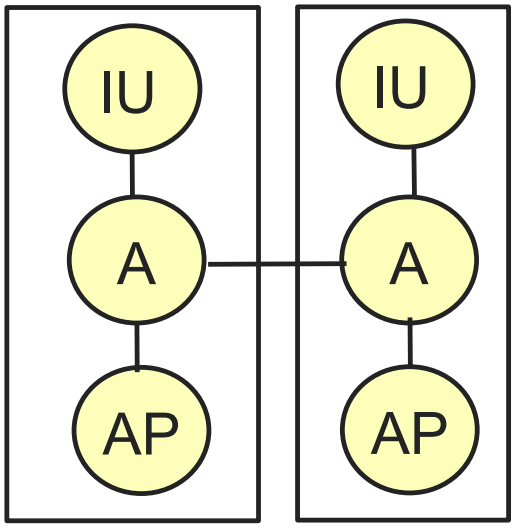


Figura 3.5: Arquitectura distribuída replicada

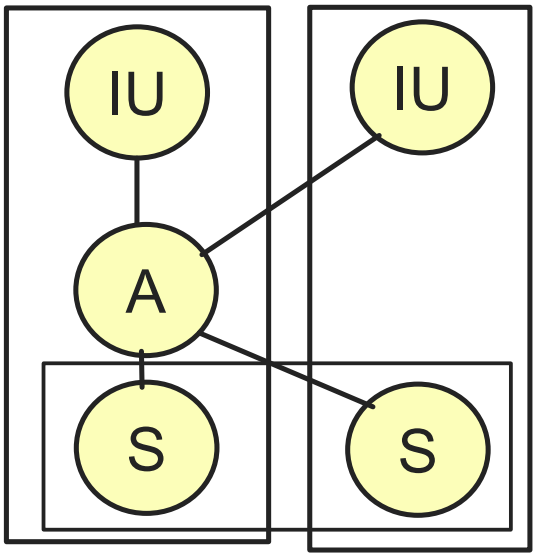


Figura 3.6: Arquitectura híbrida

agente por nó do sistema, criando uma hierarquia de agentes.

Em contrapartida, a arquitectura distribuída introduz uma complexidade adicional: a necessidade de manter a coerência dos dados que se encontram replicados no sistema. Este assunto será discutido mais adiante.

3.4 Arquitectura distribuída replicada

Numa arquitectura distribuída replicada o sistema mantêm réplicas da aplicação nos diversos nós (Greenberg e Roseman, 1996; Santos e Marcos, 1993). Para o sistema funcionar correctamente, os dados que se encontram nas diversas réplicas (AP) têm que ser mantidos coerentes (Figura 3.5).

A vantagem desta alternativa face à arquitectura distribuída pura é que o agente pode ser parte da própria aplicação, sendo apenas necessário realizar uma biblioteca muito simples para difusão de mensagens. Ao contrário, a solução distribuída pura recorre tipicamente a plataformas complexas de suporte à replicação e migração de objectos, que evidentemente oferecem funcionalidade adicional mas que podem não ser necessárias a uma aplicação cooperativa.

3.5 Arquitectura híbrida

A arquitectura híbrida – simultaneamente centralizada e distribuída – permite combinar as vantagens das arquitecturas centralizada e distribuídas

(Greenberg e Roseman, 1996). Por um lado, a existência de um agente que centraliza a difusão de informação pelos nós cliente permite gerir facilmente a utilização concorrente da aplicação e centralizar a gestão de conflitos no acesso aos dados replicados.

Por outro lado, a arquitectura híbrida mantém as vantagens que decorrem da replicação de dados, em particular a que decorre do tempo de resposta às entradas dos utilizadores.

3.6 Exemplos

NGTool

O NGTool (Antunes, 1998; Antunes e Guimarães, 1996) é um sistema de reuniões electrónicas remotas que pretende replicar algumas das características das reuniões face-a-face, designadamente a capacidade de os participantes numa reunião trocarem informação não persistente e estruturarem informação persistente. O NGTool foi construído sobre uma plataforma de suporte a trabalho cooperativo que define propriedades genéricas para todos os objectos como os quais os utilizadores do sistema interagem (visibilidade, persistência).

Esta plataforma suporta simultaneamente as noções de espaço público e privado (discutidos mais adiante). Tanto os espaços públicos como privados são áreas rectangulares não sobrepostas e contíguas (Figura 3.7) onde os utilizadores podem manipular objectos típicos de reuniões electrónicas (opiniões, comentários, decisões, etc).

Uma propriedade básica da plataforma define que os objectos movimentados entre os espaços públicos e privados adquirem funcionalidades de acordo com os espaços onde passam a residir.

Um objecto presente num espaço privado é observado e manipulado em exclusivo pelo utilizador desse espaço. O espaço público destina-se a fornecer uma vista comum sobre os objectos nele residentes a todos os seus utilizadores. Esta propriedade é implementada pela plataforma através da replicação de objectos: cada utilizador observa no espaço público uma réplica do objecto. Sendo assim, um espaço público com n utilizadores e p objectos requer a manipulação pela plataforma de n x p réplicas.

Uma outra propriedade da plataforma define que a coerência dos objectos residentes no espaço público é mantida pelo estabelecimento de acordos entre todas as réplicas, referentes às acções realizadas localmente pelos utilizadores. O exemplo típico é o do acordo que é necessário estabelecer

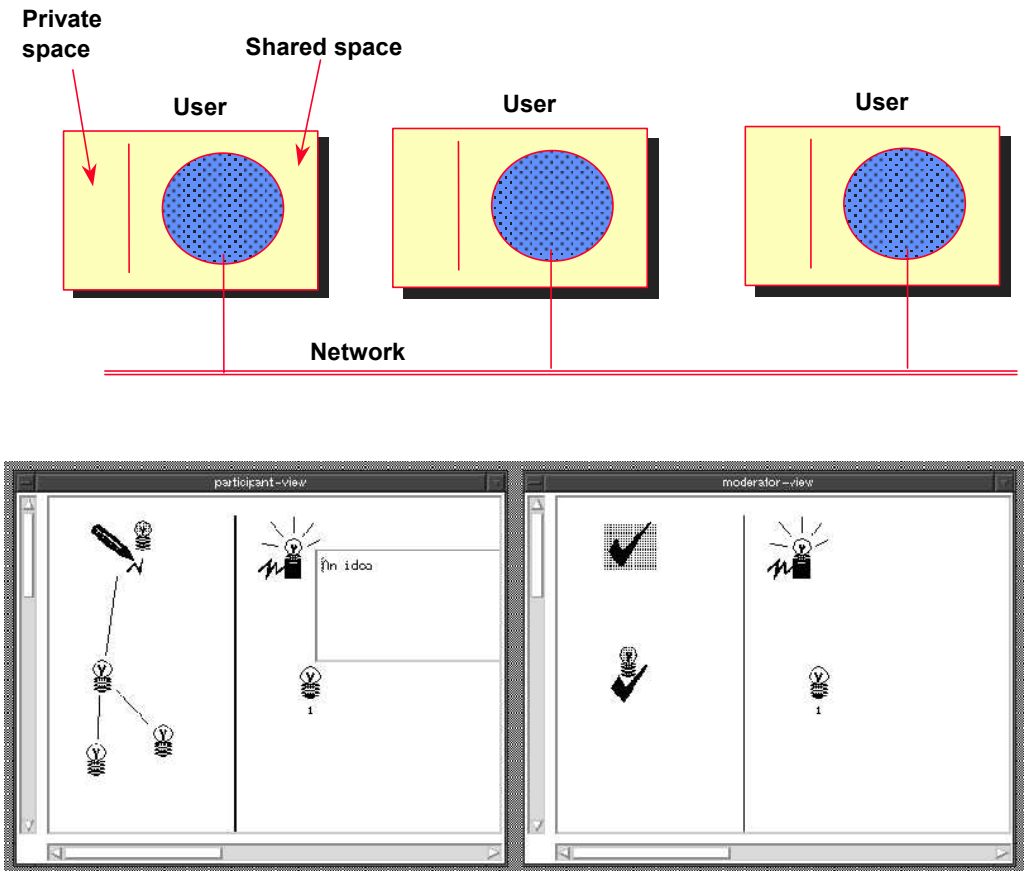


Figura 3.7: Arquitectura e interface com o utilizador do NGTool

quando um utilizador apaga um objecto que outro utilizador está a modificar.

Destes acordos resultam acções globais, difundidas por todas as réplicas. Todos os mecanismos de comunicação necessários ao estabelecimento de acordos entre réplicas e difusão de acções globais foram agregados num serviço denominado Serviço Abstracto de Comunicação.

O Serviço Abstracto de Comunicação é um servidor com o qual diversos clientes (réplicas da aplicação NGTool) estabelecem ligações TCP/IP. O Serviço Abstracto de Comunicação agrega o conjunto de funções que permitem a disseminação de informação entre as réplicas de objectos presentes em cada cliente.

No Serviço Abstracto de Comunicação os objectos replicados são apenas conhecidos pelo seu nome e posição no espaço público. As mensagens trocadas através deste serviço têm o formato definido no Quadro 3.1.

```
([endereços][mensagem])

[endereços] especifica a origem e destino das
mensagens trocadas entre réplicas de objectos e
pode ter os seguintes formatos:

 cliente1.cliente2      Mensagem para um cliente
                        específico

 cliente.TODOS          Mensagem para todos os
                        clientes

 cliente1.EXC.cliente2  Mensagem para todos os
                        clientes excepto cliente2

 SISTEMA.cliente        Mensagem com origem no
                        sistema de comunicação

[mensagem]especifica o conteúdo das mensagens
trocadas entre as réplicas, que podem ter diversos
formatos:

[Criar|Mover] objecto x y      Cria ou move
objecto

Apagar objecto                 Apaga objecto

Modificar objecto "atributo"   Modifica o objecto
```

Quadro 3.1: Tipos de mensagens trocadas através do serviço Abstracto de Comunicação

Observe-se que a arquitectura do sistema NGTool é uma arquitectura híbrida, pois o serviço de difusão de mensagens é centralizado, do tipo cliente-servidor, e os clientes deste serviço são réplicas da aplicação NGTool.

Brainstorm

Brainstorm (Barrocas, 1999) é uma ferramenta para geração de ideias baseada no conhecido método criado por Alex Osborn em 1938 (Osborn, 1963).

A ferramenta foi desenvolvida na linguagem de programação Java, versão 1.2. A escolha da linguagem deveu-se a razões de independência da plataforma, permitindo utilizar a aplicação a partir de qualquer *browser* que suporte Java.

A interface com o utilizador (Figura 3.8) foi implementada usando JFC (*Java Fundation Classes*), que é um dos componentes do JDK 1.2, também conhecido por *Swing*. Este pacote contém uma enorme variedade de objectos gráficos para a criação de interfaces com os utilizadores.

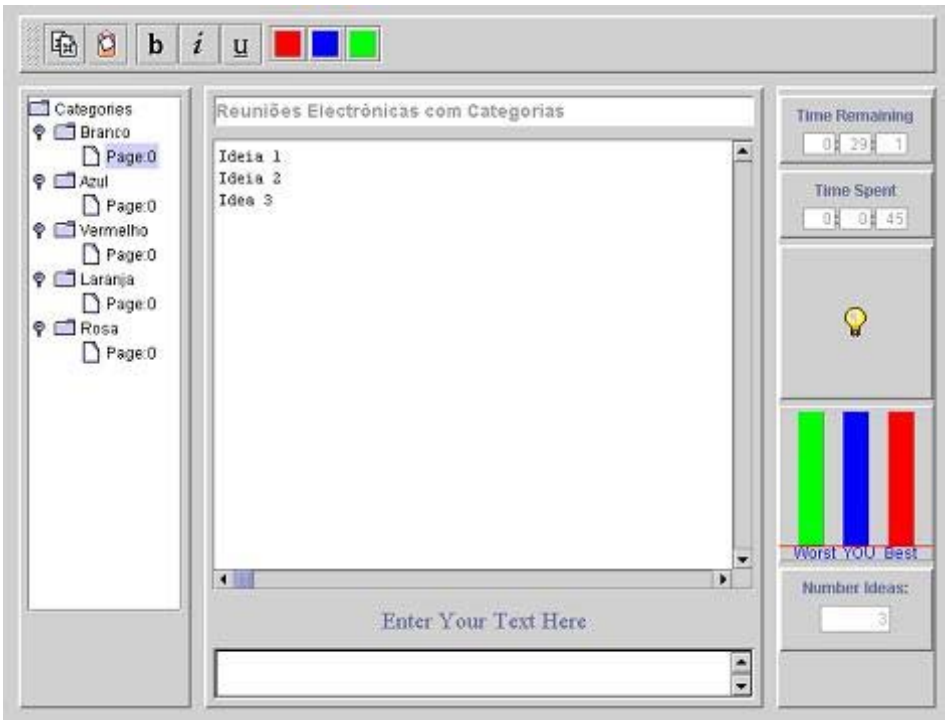


Figura 3.8: Interface com o utilizador da ferramenta Brainstorm

O sistema utiliza uma arquitectura centralizada multi-utilizador, do tipo cliente-servidor, sendo a comunicação entre servidor e clientes feita por RMI (*Remote Method Invocation*) e servidor de HTTP.

A comunicação entre cliente e servidor é efectuada em duas fases (Figura 3.9). A primeira fase consiste na selecção e configuração da sessão, dado que o Brainstorm suporta múltiplas sessões cooperativas. Nesta fase a comunicação com o servidor é feita através de um *servlet*. O *servlet* é acedido via servidor de HTTP.

O *servlet* disponibiliza informação sobre as sessões estabelecidas e todas as

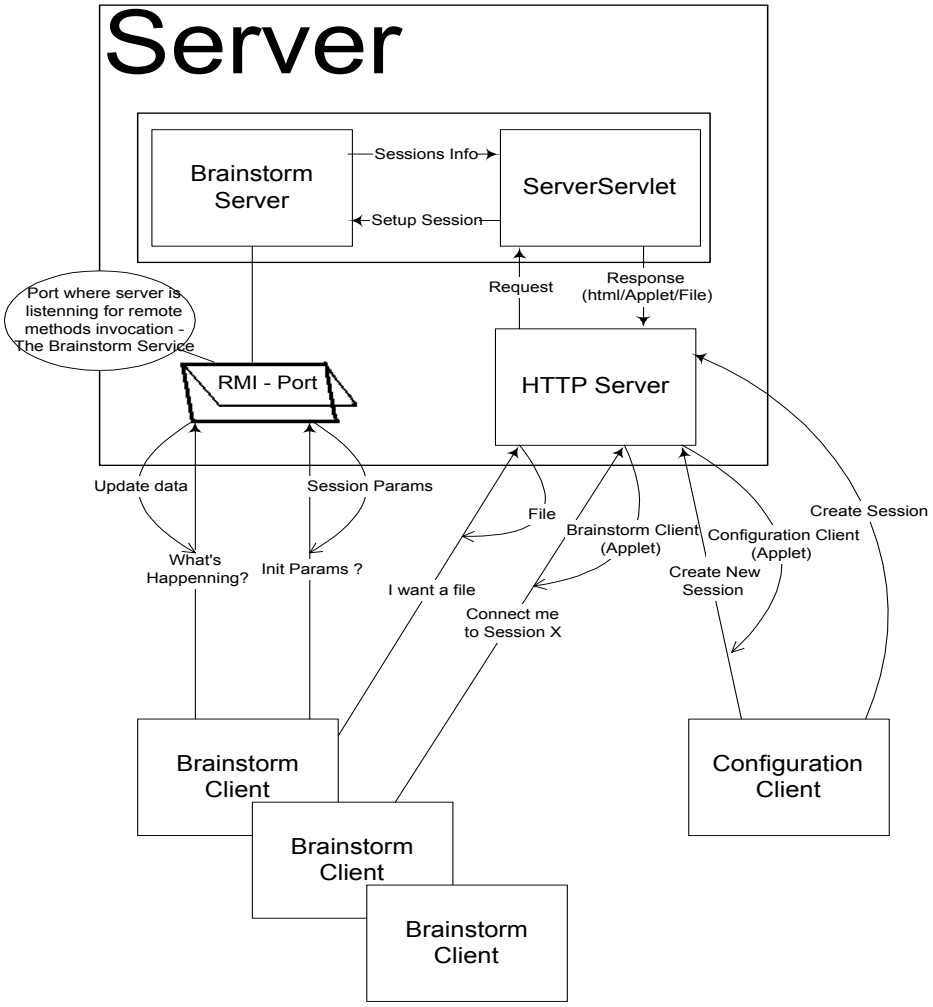


Figura 3.9: Arquitectura da ferramenta Brainstorm

funções que podem ser executadas sobre estas.

A segunda fase dá-se após a ligação do cliente a uma das sessões existentes. A partir desse momento, a comunicação entre o cliente e o servidor é efectuada através de RMI, excepto para o carregamento de ficheiros, caso em que se recorre ao *servlet*.

4 Concorrência

A gestão da concorrência coloca duas questões complementares:

- Nas arquitecturas que suportam concorrência, torna-se necessário controlar o acesso dos diversos utilizadores do sistema aos dados partilhados.
- Nas arquitecturas que, para além da concorrência, suportam a replicação de dados é também necessário controlar a coerência das réplicas.

4.1 Controlo da concorrência

O controlo da concorrência é efectuado através de mecanismos de controlo da concorrência. Estes mecanismos podem recorrer a políticas pessimistas, optimistas ou fragmentadas.

Política pessimista

Uma política pessimista de controlo da concorrência advoga a verificação e resolução de conflitos pelo sistema de forma a anular a possibilidade de a coerência de dados ser comprometida. Diversos mecanismos de controlo da concorrência adoptam esta política (Ellis et al., 1991):

- Trinco (*locking*) – O acesso aos dados é gerido de forma a que apenas um utilizador – o detentor do trinco – possa aceder aos dados.

  Esta política é bastante restritiva, dado que, à excepção do detentor do trinco, os restantes utilizadores ficam impedidos de aceder aos

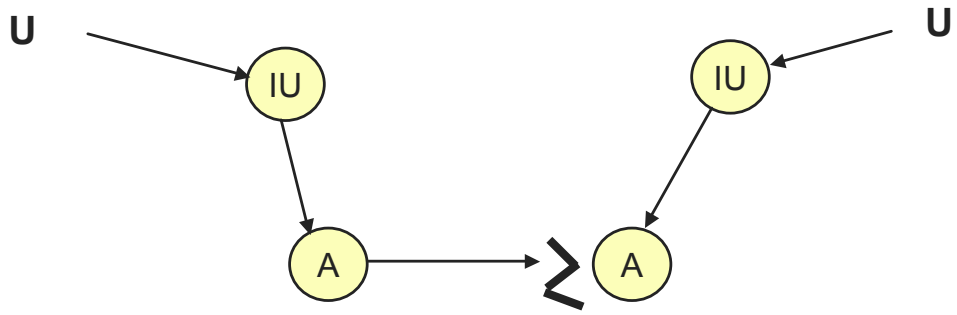


Figura 4.1: Ilustração do mecanismo de trinco: O detentor do trinco nega o acesso aos outros clientes

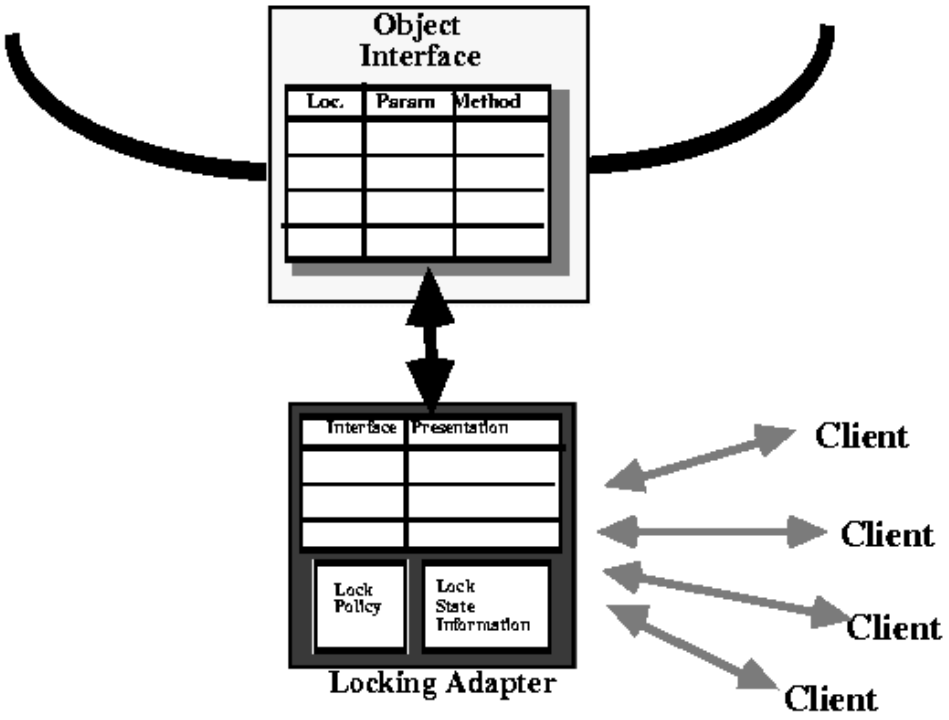
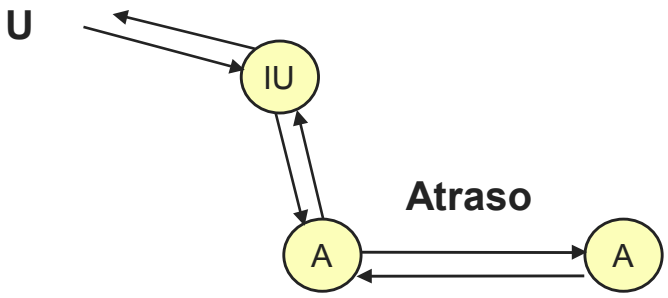


Figura 4.2: Realização de um mecanismo de múltiplos trincos



dados partilhados (Figura 4.1).

- Múltiplos trincos – Trata-se de um mecanismo semelhante ao anterior, onde o trinco não se aplica globalmente a toda a informação partilhada pela aplicação mas antes ao nível do objecto, de modo a permitir actividades paralelas dos utilizadores (Figura 4.2). Esta

abordagem permite reduzir a probabilidade de os utilizadores ficarem impedidos de aceder aos dados.

- Trincos móveis (*roving-locks*) – Um dos problemas fundamentais associados ao uso de trincos é que, num sistema sujeito a elevados atrasos na comunicação entre os seus nós, a obtenção do trinco pode ser demasiado demorada, levando à rejeição do sistema pelo utilizador (Figura 4.3).

  O mecanismo de trincos móveis tenta reduzir o atraso na obtenção de um trinco: nesta abordagem cada utilizador pede ao sistema para lhe atribuir, em antecipação, trincos para um conjunto de dados cuja probabilidade de utilizar no futuro seja elevada.

- Transacções – Esta é uma solução tradicional das bases de dados que permite a manipulação de dados em unidades atómicas (transacções), garantindo que a informação distribuída sofre sempre transformações coerentes. Existem múltiplas variantes desta abordagem.

  Algumas destas variantes utilizam o conceito de controlo optimista da concorrência (cada transacção tem duas ou três fases, sendo a primeira não bloqueante; Barghouti e Kaiser, 1991). Note-se no entanto que este conceito de optimisto utilizado na área das bases de

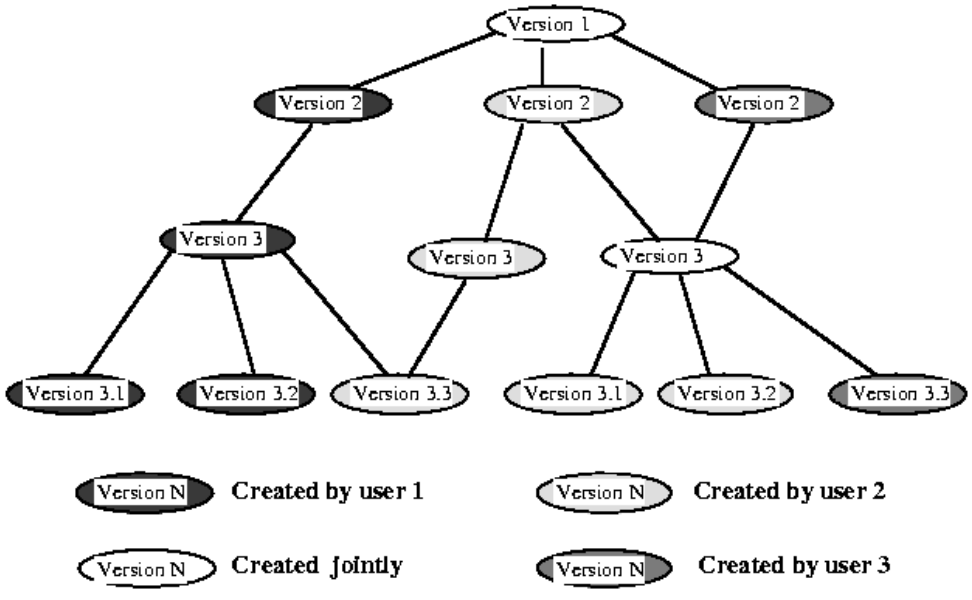


Figura 4.4: Mecanismo de versões

dados é diferente do conceito de optimisto utilizado na área de groupware já que, apesar de considerar uma fase optimista, as restantes fases são pessimistas.

No caso dos sistemas cooperativos, o uso de transacções pode ainda ser problemático porque, ao contrário dos sistemas tradicionais, as transacções são geralmente de longa duração. Existem no entanto variantes do mecanismo de transacções que suportam transacções longas. Deve dizer-se, no entanto, que mesmo estas variantes tendem a ser pouco utilizadas, essencialmente por duas razões: (1) as transacções muito longas conduzem a uma menor colaboração entre os utilizadores; (2) geralmente as aplicações cooperativas, por fornecerem mecanismos de interacção entre os seus utilizadores, não necessitam de mecanismos tão estritos de controlo da concorrência.

- Versões – O sistema inspecciona o acesso aos objectos e, ao detectar um conflito, cria uma nova versão do objecto, prosseguindo com actividades paralelas (Figura 4.4). Os problema associados a esta alternativa é que, por um lado, requer mecanismos de reconciliação de versões e, por outro lado, conduz igualmente a uma menor a colaboração entre os utilizadores (se a reconciliação não for frequente).
- Transacções conversacionais – Este é um mecanismo que integra os mecanismos de controlo de versões e transacções descritos acima de modo a introduzir primitivas de controlo de versões (Barghouti e Kaiser, 1991).

Política optimista

A adopção de uma política pessimista para gestão da informação condiciona significativamente a funcionalidade global de um sistema cooperativo, pois impôem uma de três alternativas:

- Restringir as interacções dos utilizadores, não permitindo a manipulação da mesma informação em paralelo sem antes ser obtido um acordo explícito.
- Permitir a manipulação da mesma informação pelos utilizadores, sujeita a protocolos de acesso aos dados, que por sua vez requerem comunicação entre os nós do sistema e que portanto podem introduzir atrasos inaceitáveis pelos utilizadores do sistema (Stefik et al., 1987; Barghouti e Kaiser, 1991).
- Não sujeitar os utilizadores a restrições ou protocolos de acesso aos dados mas, em contrapartida, restringindo a cooperatividade entre os utilizadores do sistema.

A política optimista destina-se a obviar os problemas acima indicados. No entanto, é introduzido um custo adicional no sistema: a detecção e resolução de conflitos não é garantida pelo sistema, torna-se explícita para os utilizadores, que devem dispender um esforço adicional para manter a coerência dos dados.

A política optimista de controlo da concorrência permite que os dados partilhados sejam livremente manipulados pelos utilizadores. A verificação e possível resolução dos conflitos é realizada posteriormente. Diversos tipos de mecanismos adoptam esta política:

- Mecanismos cooperativos – Não é fornecido qualquer controlo da concorrência, nem verificação ou resolução de conflitos (Stefik et al., 1987).

  Sendo uma opção arrojada, a ideia por detrás destes mecanismos consiste em tornar as alterações de dados visíveis a todos os utilizadores, de modo a que sejam estes a verificar os possíveis conflitos e resolvê-los cooperativamente.

- Detecção de dependências – Este é um tipo de mecanismo que adiciona aos mecanismos cooperativos a detecção de conflitos. O sistema, após detectar um conflito, informa os utilizadores da necessidade de o resolverem cooperativamente.
- Transformação de operações – Recorrem a uma estratégia em que o sistema fornece aos utilizadores a possibilidade de, após detectarem um conflito, reporem o estado anterior do sistema, transformando as operações executadas. As incoerências de dados existem assim apenas temporariamente no sistema. As transformações correspondem a operações inversas às que forem executadas sobre os dados: adicionar, apagar, etc.

Política fragmentada

A política fragmentada optimiza as características das políticas pessimista e optimista, seja para um conjunto de utilizadores, seja para um conjunto de objectos do sistema.

A política fragmentada associa assim propriedades das políticas pessimista e optimista. Por exemplo, alguns sistemas permitem que sejam os utilizadores a definir qual o mecanismo de controlo da concorrência a utilizar em cada objecto (Pacull et al., 1994). Os utilizadores podem igualmente definir a granularidade dos objectos.

Barghouti e Kaiser (1991) descrevem uma outra aproximação que divide hierarquicamente o controlo da concorrência por pequenos grupos de utilizadores, recorrendo nesse caso preferencialmente a uma política optimista; e o conjunto global dos diversos grupos de utilizadores, onde a política seleccionada é pessimista.

4.2 Controlo das réplicas

A questão do controlo das réplicas surge nas arquitecturas distribuídas (puras, replicadas ou híbridas) e pode estar ou não interligada com o controlo da concorrência. Essencialmente podem considerar-se as seguintes alternativas para controlo das réplicas:

- Delegar o controlo no sistema, definindo um protocolo a ser seguido por este. Por exemplo, os mecanismos de *proxy* e de *cache* definem protocolos simples que tentam optimizar a criação e leitura/escrita efectuadas sobre as réplicas.
- Delegar o controlo nos utilizadores. Nestes casos o sistema apenas fornece meios de comunicação que permitem aos utilizadores desenvolverem protocolos sociais para resolução de conflitos.
- Delegar o controlo numa entidade do sistema que fica explicitamente responsável por interagir com os utilizadores de modo a resolver os conflitos. Nestes casos o sistema combina as duas funcionalidades acima, procurando resolver os conflitos de forma dinâmica, flexível e negociada (Simone et al., 1999).

4.3 Questão da transparência

Consideremos de novo um sistema cooperativo com uma arquitectura distribuída (pura, replicada ou híbrida). Este sistema pode ser concebido como sendo constituído por três camadas distintas (Figura 4.5):

- IU – Camada de interface com os utilizadores, tipicamente responsável pelos aspectos de apresentação da informação e interacção com o utilizador;
- S – Camada contendo a semântica da aplicação;
- A – Camada de *middleware*, responsável pela comunicação entre os nós do sistema e também pelo controlo da concorrência.

Face a este cenário, uma questão que se pode colocar é a seguinte: até que ponto deve a camada de *middleware* esconder os mecanismos e políticas de controlo da concorrência?

Se a camada de *middleware* mascarar completamente esses mecanismos e políticas, o programador do sistema cooperativo obtém uma série de vantagens, como sejam:

- Contratualização com a camada de *middleware* de uma série de garantias como a entrega fiável das mensagens, a ordem das mensagens ou tolerância a faltas.

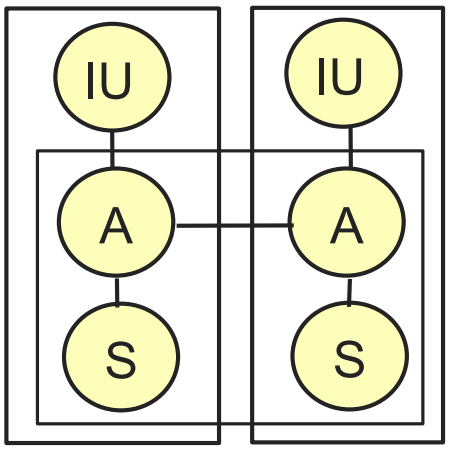


Figura 4.5: Sistema cooperativo constituído por três camadas

- Utilização de uma API (*Application Interface*) simples para acesso à camada de *middleware*.
- Simplicidade de programação, que decorre directamente dos dois aspectos indicados acima.

Uma camada de *middleware* que oferece estas vantagens pode ser considerada transparente: tem uma simplicidade aparente, sendo a sua complexidade escondida pela API.

Infelizmente, a transparência traz também um conjunto significativo de desvantagens para o programador do sistema cooperativo:

- O sistema torna-se rígido, no sentido em que, por vezes, não é possível modificar dinamicamente os mecanismos e políticas de controlo da concorrência. Por exemplo, mudando de uma situação optimista para uma pessimista quando ocorrem muitos conflitos.
- As garantias dadas pela camada de *middleware* não levam em conta muitos aspectos da semântica da aplicação. Por exemplo, algumas aplicações cooperativas têm utilizadores mais importantes do que outros, podendo estes últimos ser excluídos do sistema em situações de elevado tempo de resposta.

- Os utilizadores não são informados das decisões tomadas pela camada de *middleware.*

Por estas razões os programadores de sistemas cooperativos tendem a evitar o recursos a camadas do sistema transparentes. Note-se, no entanto, que a questão da transparência também preocupa a comunidade que se dedica ao desenvolvimento de *middleware*, sendo expectáveis a curto prazo soluções para esse problema (que passam pela negociação flexível de políticas, arquitecturas de *middleware* reflexivas e acesso externo à estrutura interna do *middleware*; Tripathi, 2002; Kon et al., 2002).

4.4 Exemplos

GroupSystems

O sistema GroupSystems (Nunamaker et al., 1991) fornece uma ferramenta de geração de ideias que utiliza uma técnica denominada *brainwriting* (uma variante da técnica *brainstorming*; Hwang e Lin, 1987). Esta técnica

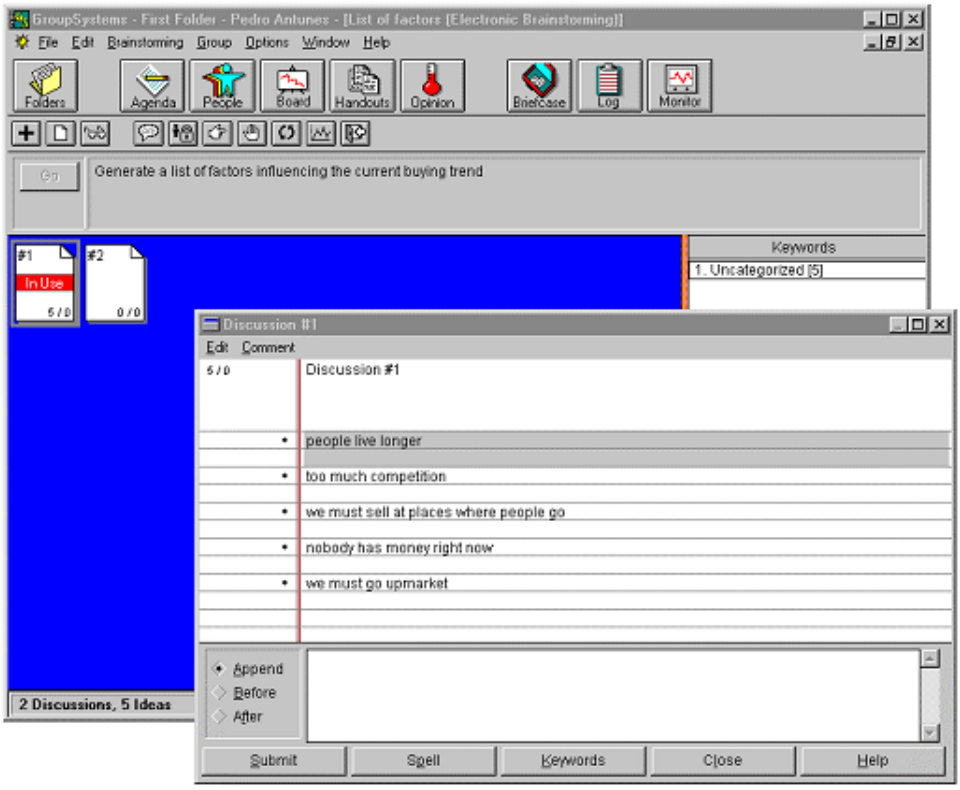


Figura 4.6: Ferramenta de geração de ideias do sistema GroupSystems. Observe-se a utilização do mecanismo de trinco (Fonte: www.groupsystems.com)

considera a existência de um número de folhas de papel igual ao número de participantes e colocadas numa mesa diante destes (Figura 4.6). Cada participante retira uma folha da mesa, observa o seu conteúdo, acrescenta uma ideia, volta a colocar a folha na mesa e passa a outra folha. Este processo repete-se até ao esgotamento de novas ideias pelos participantes.

O sistema GroupSystems utiliza trincos para controlar o acesso dos utilizadores à versão computacional das folhas de papel.

NGTool

Como foi mencionado anteriormente, o sistema NGTool procede à partilha de informação pelos utilizadores através da replicação dos objectos públicos. Como consequência, surge, por um lado, a necessidade de gerir as diversas réplicas dos objectos públicos e, por outro lado, a necessidade de exercer um controlo da concorrência das acções dos utilizadores sobre os objectos públicos.

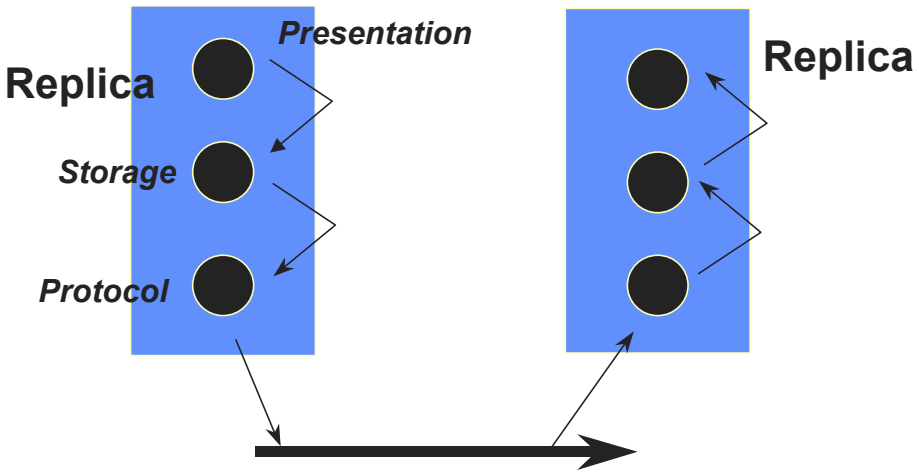


Figura 4.7: Implementação dos objectos públicos no NGTool

Internamente, o sistema implementa um objecto público a partir da composição de três outros objectos de nível inferior (Figura 4.7):

- *Storage* – Objecto de dados exclusivamente destinado a armazenar a informação do objecto público (ideia, comentário, etc).
- *Presentation* – Objecto gráfico cuja função é mediar o acesso do utilizador aos dados armazenados no *storage*.
- *Protocol* – Objecto que define a política e exerce o controlo das réplicas do objecto público.

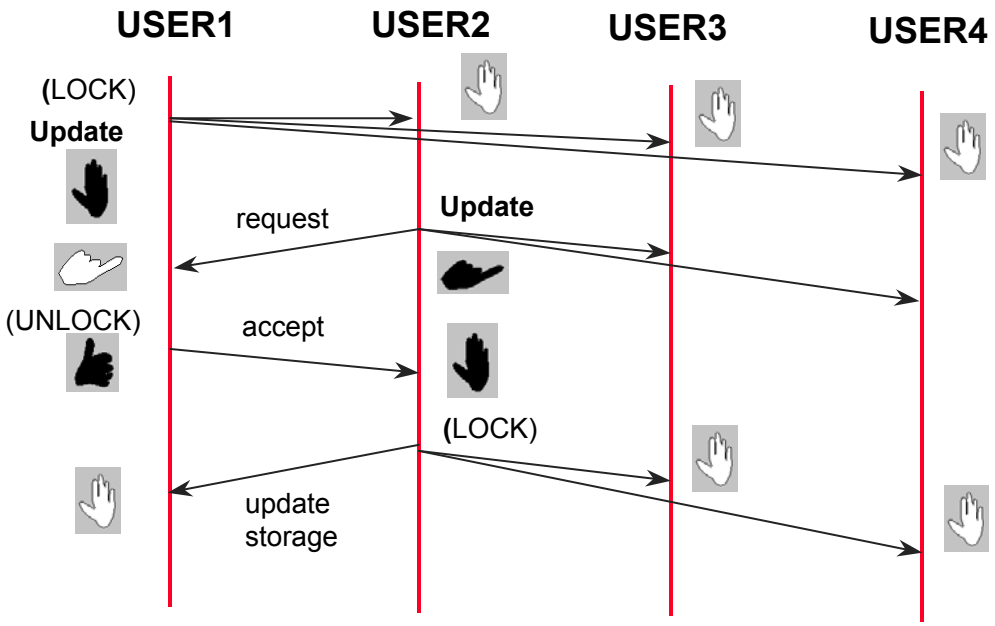


Figura 4.8: Protocolo de trinco implementado pelo NGTool (as figuras icónicas aparecem junto ao objecto público para indicar aos utilizadores o desenrolar do protocolo)

Todos estes objectos de nível inferior são replicados pela plataforma do sistema. Para exercer a sua função, as réplicas do *protocol* necessitam de trocar mensagens através do Serviço Abstracto de Comunicação e de actuar localmente sobre os *storage* de cada réplica.

Mais concretamente, um *protocol* pode executar as seguintes acções: (1) enviar mensagens a outras réplicas; (2) receber mensagens de outras réplicas; (3) mover a *presentation* no espaço público; (4) modificar o conteúdo do *storage*; (5) eliminar o objecto público.

As mensagens trocada entre réplicas de um *protocol* dependem do mecanismo de controlo que for seleccionado pelo programador. No caso do *protocol* de trinco, um único utilizador – o detentor do trinco – pode modificar o *storage*.

Na situação inicial apresentada na Figura 4.8, USER 1 é o detentor do trinco. USER 2, pretendendo alterar os dados do objecto, requisita o trinco, pelo que a réplica do *protocol* difunde a mensagem *request*. A réplica do *protocol* associada ao USER 1 recebe o pedido e, quando possível, liberta o trinco enviando a mensagem *accept*. USER 2 passa a detentor do trinco e modifica o conteúdo do *storage*, difundindo a todas as réplicas a mensagem

update storage (na ocorrência de faltas, este protocolo torna-se mais complexo; Cosquer et al., 1996).

O facto de o sistema NGTool recorrer a uma camada de *middleware* não transparente (o Serviço Abstracto de Comunicação), permite notificar os utilizadores do sistema cooperativo quer sobre o andamento do protocolo quer sobre os atrasos que ocorrem na comunicação.

O NGTool recebe mensagens do Serviço Abstracto de Comunicação informando sobre a progressão (do sistema na difusão) de uma acção pelas diversas réplicas de um objecto, dando ao utilizador uma noção dinâmica do funcionamento do sistema.

A Figura 4.9 ilustra a troca adicional de mensagens entre o Serviço Abstracto de Comunicação e a interface com o utilizador: inicialmente USER 1 aplica uma acção local sobre a réplica de um objecto; em seguida, o Serviço Abstracto de Comunicação comunica com as restantes réplicas do objecto; em seguida são enviadas mensagens de resposta por cada réplica que é actualizada; finalmente, são coleccionadas as respostas correspondentes à difusão da acção local pelas diversas réplicas.

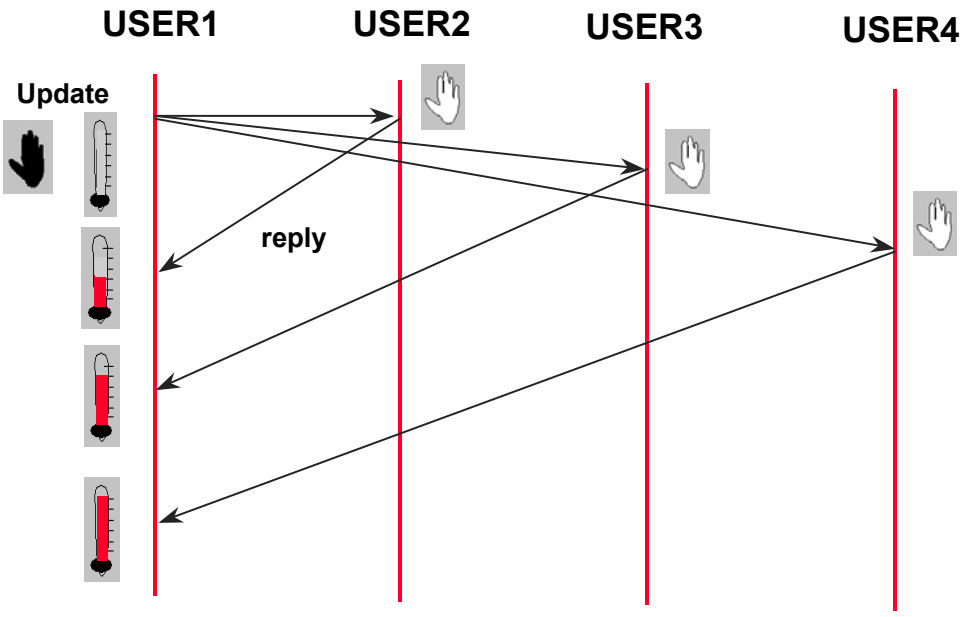


Figura 4.9: Troca adicional de mensagens destinadas a informar os utilizadores sobre a progressão do sistema quanto à coerência dos dados

As indicações fornecidas aos utilizadores do NGTool reflectem assim a percentagem de utilizadores que observam correctamente um objecto público.

Note-se que esta funcionalidade apenas se justifica a partir de um determinado tempo de resposta do sistema.

IdeaGen

Tal como o Brainstorm, o IdeaGen (Alves e Silva, 1997) é uma aplicação de geração de ideias. A geração de ideias é realizada num espaço privado de cada utilizador.

A estruturação das ideias obedece a uma forma hierárquica: no topo da hierarquia existe obrigatoriamente uma "pasta" de ideias e em níveis inferiores podem existir ideias ou outras "pastas" com ideias. Esta aproximação assemelha-se à estrutura hierárquica de um editor de texto com diversas secções, sub-secções e parágrafos, ou à estrutura de um sistema de ficheiros com os seus directórios e sub directórios.

No IdeaGen, a estruturação cooperativa de ideias é conseguida através da possibilidade de, cooperativamente, definir e organizar "pastas" de ideias, o equivalente num editor de texto, a escrever uma secção principal e uma subsecção em paralelo para no fim as integrar. O trabalho em paralelo fica favorecido por esta forma de estruturação, pois a fragmentação hierárquica reduz a probabilidade de conflito ao mesmo tempo que integra naturalmente a informação produzida.

Após a geração privada das ideias, o participante tem a possibilidade de tornar a sua estrutura de ideias visível aos restantes elementos do grupo, bastando para tal colocá-la no espaço público. Neste espaço todos os participantes podem editar os conteúdos, alterar a estruturação das ideias e as dependências entre "pastas" de ideias.

A arquitectura da aplicação é do tipo distribuída híbrida (Figura 4.10). O controlo de cópias é realizado por um nó central, localizado num servidor denominado IdeaServer, onde são analisados os conflitos e geridos os mecanismos de recuperação.

No entanto, o nó de controlo não utiliza uma política pessimista: apesar de toda a comunicação entre réplicas dos objectos passar pelo nó de controlo, continua-se a manter a comunicação distribuída de modo a gerar o paralelismo.

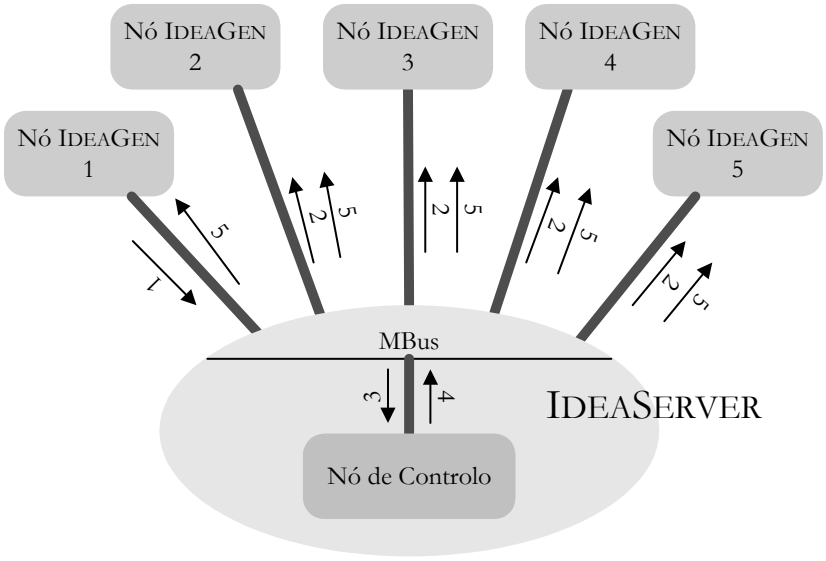


Figura 4.10: Política optimista: O nó 1 difunde a sua mensagem não só para o nó de controlo como também para as restantes réplicas do sistema

As operações efectuadas sobre os objectos públicos são comunicadas a todos os restantes nós, incluindo o de controlo. Na ocorrência de conflitos, o nó de controlo encarrega-se de comunicá-los a todos os nós, que os deverão resolver posteriormente.

A política optimista adoptada desenrola-se em cinco fases:

- Um nó (na Figura 4.10 o nó 1) envia ao servidor IdeaServer um evento para distribuir pelas réplicas.
- O evento é enviado para os nós seleccionados na mensagem (na figura, todos os restantes).
- O evento é igualmente comunicado ao nó de controlo residente no servidor.
- Caso seja detectado um conflito, o nó de controlo envia uma notificação ao serviço de distribuição do IdeaServer.
- O serviço distribui o evento por todos os nós que, por sua vez, tratam de informar o utilizador.

Depois de os nós do sistema serem informados da ocorrência de um conflito, entra em funcionamento um mecanismo de reconciliação. Este mecanismo atribui um dos seguintes estados a cada objecto público:

- Livre – Não existem conflitos. Apenas um utilizador está a aceder ao objecto.

Figura 4.11: Ilustração do mecanismo de reconciliação

- Conflitos – Detectaram-se conflitos. Mais do que um utilizador está a editar o objecto. Informa-se o utilizador da sua ocorrência.
- Em Solução – Os conflitos devem ser resolvidos pelo que não são permitidas alterações ao conteúdo do objecto público. Proíbe-se a edição da réplica local e promove-se a votação pelos utilizadores envolvidos no conflito de qual o conteúdo que deve ser atribuído ao objecto público.

A informação fornecida ao utilizador pelo mecanismo de reconciliação utiliza o paradigma do semáforo (Figura 4.11): amarelo indica a ocorrência de conflitos e vermelho indica a proibição de edição. Apresenta-se na Figura 4.12 o funcionamento do mecanismo de reconciliação no IdeaGen.

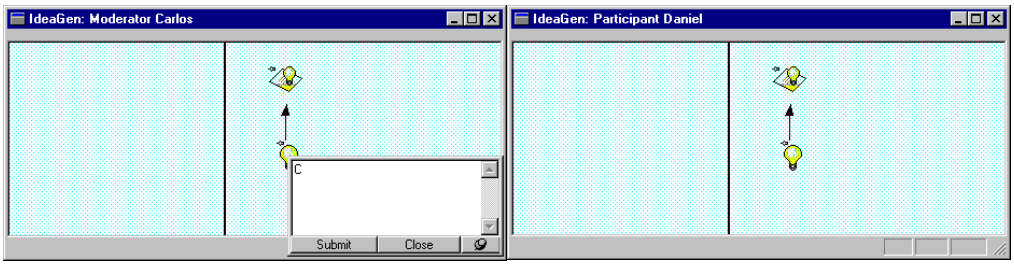


Figura 4.12a: O primeiro utilizador inicia a edição do objecto público

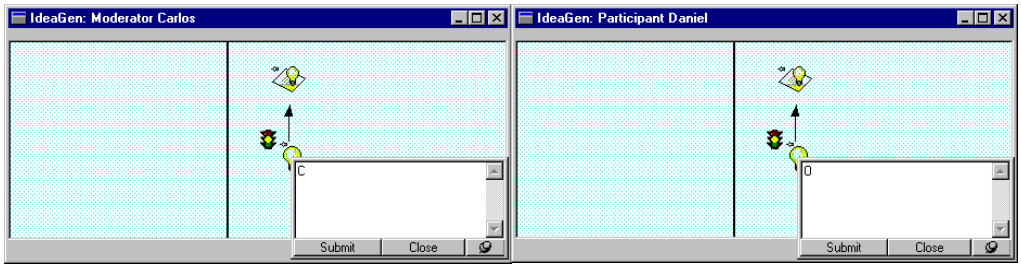


Figura 4.12b: O segundo utilizador inicia também a edição do mesmo objecto público. O conflito é detectado e os utilizadores são informados pelo semáforo amarelo

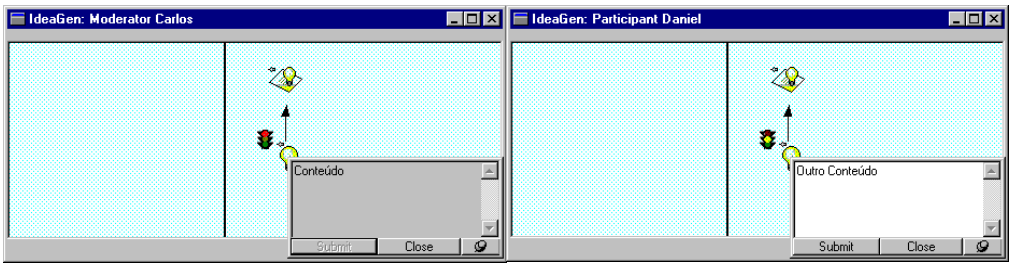


Figura 4.12c: O primeiro utilizador submete a alteração de dados do objecto público. Deixa de poder editar a réplica e o semáforo passa a vermelho

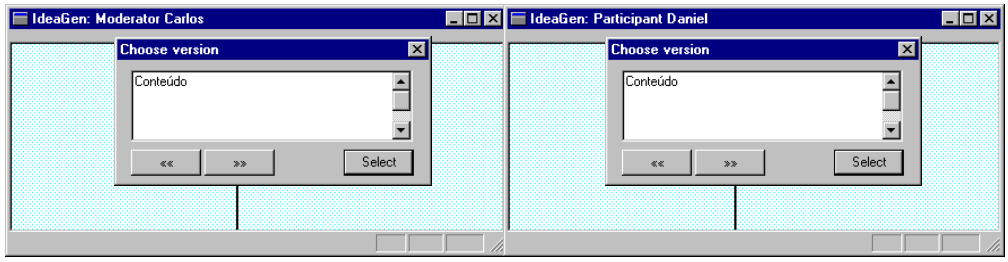


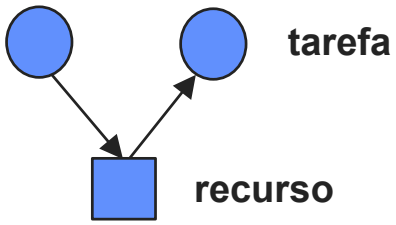
Figura 4.12d: O segundo utilizador submeteu também uma alteração de dados. Ninguém se encontra a editar pelo que se inicia a escolha de versão. Ambos os utilizadores votam numa versão. A versão mais votada será adoptada. O semáforo passa a verde por momentos, desaparecendo depois. O mecanismo de reconciliação terminou a sua função

5 Coordenação

A complexidade dos problemas acaba sempre por determinar o recurso a uma estratégia de partição do trabalho de grupo em actividades individuais ou a realizar por sub-grupos. Esta estratégia de organização tem como objectivo optimizar a produtividade global, aproveitando as capacidades individuais, evitando tempos mortos e reduzindo os custos de comunicação associados ao trabalho de grupo. Em contrapartida aos custos de comunicação, a partição do trabalho requer alguma forma de coordenação de actividades, o que também apresenta custos (decorrentes de redundâncias, atrazos e por vezes repetições de actividades; Whittaker e Schwarz, 1999).

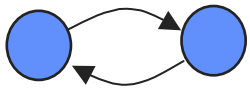
Iremos aqui apenas abordar os diferentes mecanismos computacionais de suporte a coordenação de actividades no âmbito da cooperação. Genericamente, podem-se considerar três tipos fundamentais de coordenação (Figura 5.1; malone e Crowston, 1994; Butler, 1991):

- Indirecta – Quando os elementos do grupo partilham um determinado artefacto que serve de mediador entre os participantes, que efectuam transacções sobre ele de acordo com regras pré-estabelecidas.
- Sequencial – Quando os elementos do grupo estabelecem um fluxo de



Coordenação indirecta

Coordenação sequencial



Coordenação recíproca

Figura 5.1: Tipos fundamentais de coordenação

actividades que, sequencialmente, passam de indivíduo para indivíduo. O desencadear de uma determinada actividade depende do fim da actividade anterior. Existe normalmente um plano ou um conjunto de procedimentos que definem a priori a sequência de actividades e permitem coordenar as actividades individuais.

- Recíproca – Quando as actividades para serem realizadas necessitam de fluxos de informação nos dois sentidos, entre os indivíduos, num processo de ajustamento mútuo (Mintzberg, 1993).

Colocada a questão nestes termos, pode dizer-se que a coordenação indirecta foi já abordada na secção anterior. No essencial, a selecção de mecanismos e políticas de controlo da concorrência serve de suporte a coordenação indirecta. Assim, iremos em seguida apenas identificar mecanismos de suporte a coordenação sequencial e recíproca.

5.1 Coordenação sequencial formal

O primeiro tipo de mecanismo de coordenação aqui apresentado é o que efectivamente se encontra mais enraizado nas tecnologias de informação: os mecanismos sequenciais formais desenvolvem-se a partir de modelos de especificação das estruturas, intervenientes e processos no domínio de aplicação. A sua execução consiste na atribuição de tarefas, gestão de actividades e controlo do desenvolvimento do processo.

Historicamente, distinguem-se duas tendências de desenvolvimento de mecanismos sequenciais formais para coordenação de tarefas: uma inicialmente voltada para a circulação de informação (*circulation folders*) em ambientes de escritório electrónico e uma tendência mais recente, voltada para os fluxos de trabalho (*workflow*) nas organizações.

Uma crítica de Ellis e Wainer (1994) sobre este tipo de mecanismos é que a formalização da coordenação leva à "automatização de uma ficção", pois os processos reais de trabalho nas organizações são constituídos por um misto de procedimentos formais e informais. Ou seja, existe a necessidade de complementar a utilização de mecanismos formais com outros, não formais, capazes de introduzir flexibilidade nos processos de trabalho em grupo.

A asserção inversa é igualmente verdadeira: os mecanismos formais de coordenação apresentam características únicas, nomeadamente a captura que fazem do funcionamento regular das organizações, pelo que se mostra vantajosa a sua integração com mecanismos não formais.

5.2 Coordenação sequencial semi-formal

A coordenação semi-formal recorre a um mecanismo de suporte à troca de mensagens semi-formais entre indivíduos.

Genericamente, distinguem-se três tipos de mensagens na comunicação suportada por computador:

- Formais – Definem a estrutura dos dados contidos nas mensagens, como por exemplo uma mensagem do tipo RPC (*Remote Procedure Call*).
- Informais – Não definem qualquer estrutura de dados.
- Semi-formais – Associam campos formais e informais numa única mensagem.

Um exemplo de mensagem semi-formal é uma mensagem de correio electrónico, onde existe um campo formal, definindo o assunto, origem e destinatário da mensagem; e um campo informal, constituído pelo texto que forma o corpo da mensagem.

A coordenação a partir de mensagens semi-formais surge da associação de tarefas computacionais à informação semântica transmitida na parte formal das mensagens. O sistema cooperativo pode assim estabelecer padrões de interacção entre utilizadores combinando as mensagens semi-formais com regras e agentes computacionais.

As mensagens semi-formais foram originalmente desenvolvidas no sistema Information Lens (Malone et al., 1987, 1993). Um exemplo comercial do mesmo conceito pode ser encontrado no Lotus Notes (Bragen, 1994).

O sistema Information Lens permite definir formulários de mensagens (do tipo "marcar reunião", "cancelar reunião", "anunciar conferência", "pedir informação", etc). Cada formulário é constituído por um conjunto de campos formais do tipo: "tópico", "data", "tema" ou "urgência". O sistema inclui editores de formulários e regras. As regras são constituídas por condições (por exemplo, se o tema é do tipo x) que, quando verdadeiras, originam manipulações simples das mensagens (apagar, mostrar, mover).

As regras podem ser locais, definindo manipulações de mensagens recebidas por um utilizador, ou centrais, definindo quais as mensagens que os utilizadores desejam receber. O sistema Information Lens sugere ainda tipos de mensagens de resposta a mensagens recebidas.

Note-se que neste tipo de sistemas cooperativos são os utilizadores que identificam o tipo de mensagens que desejam enviar e receber. Esta particularidade cria dificuldades à cooperação, pois um utilizador não sabe que utilizadores recebem determinada mensagem. Este problema, no entanto, só se verifica para grupos de pequena dimensão, onde um participante tem a expectativa de interactuar com a totalidade do grupo. Quando o número de elementos do grupo é muito elevado, a expectativa é de que a participação fique fragmentada em pequenos grupos de discussão.

O objectivo fundamental dos mecanismos semi-formais é filtrar as mensagens recebidas pelos utilizadores, pelo que a sua utilização para coordenação só se torna adequada quando se encontra envolvido um número muito elevado de utilizadores. A outras restrição associada a este sistema é que ele deve suportar à partida uma conjunto de padrões de interacção úteis para os utilizadores.

5.3 Coordenação recíproca, por controlo de palco

O mecanismo de controlo de palco coordena as actividades de uma forma semelhante à utilizada numa conferência em que os participantes se encontram face-a-face. Numa conferência os participantes não podem comunicar todos em simultâneo, existindo ao invés um palco privilegiado de onde um participante previamente seleccionado se dirige aos restantes participantes.

O palco assume portanto um papel central na coordenação. Estando definido o mecanismo básico, podem em seguida ser consideradas diversas políticas para controlo do palco:

- Centralizada – A autoridade reside num moderador que circula o acesso exclusivo ao palco pelos participantes.
- Competitiva – Cada participante concorre no acesso ao palco, ganhando, por exemplo, aquele que for mais rápido.
- Aberta (*open-floor*) – Os participantes têm livre acesso ao palco, sendo o controlo exercido socialmente, segundo regras de etiqueta.

Note-se que o mecanismo de conferência aplica uma política muito restritiva de acesso aos canais de comunicação entre os participantes, dado que, em cada momento, apenas um participante – o detentor do palco – se encontra activo no sistema.

Este tipo de mecanismo é mais adequado a sistemas tempo-igual/local-igual, pois existem canais de comunicação adicionais (visuais, auditivos) que

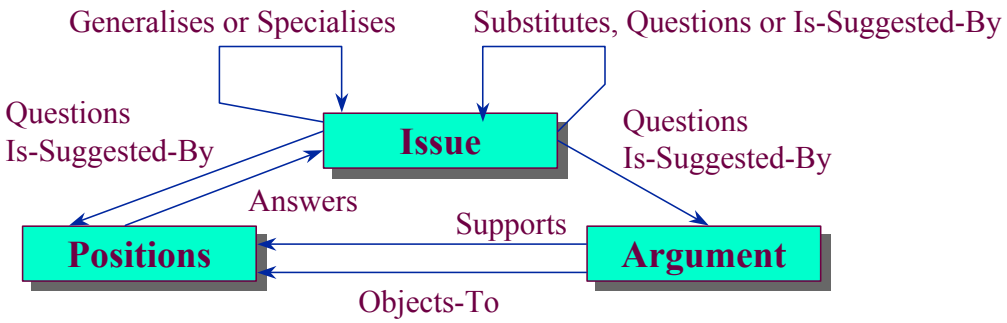


Figura 5.2: Modelo IBIS

permitem interromper, ultrapassar e complementar a gestão exercida pelo sistema.

Note-se ainda que nos sistemas tempo-igual/local-diferente o tempo necessário à comutação do palco pode tornar-se excessivamente longo, inviabilizando a utilização do sistema.

5.4 Coordenação recíproca, por mecanismo de argumentação

Os sistemas de argumentação baseiam-se frequentemente no modelo IBIS (*Issue Based Information System*; Conklin, 1988).

Este modelo define um grupo de participantes de um processo retórico onde cada protagonista pode apresentar posições e argumentos relacionados com um determinado assunto. Cada assunto pode originar várias posições que, ou resolvem o assunto, ou o contrapõem. Cada posição é sustentada por um argumento. Uma posição, por sua vez, pode dar origem a novos assuntos, gerando-se assim uma árvore de assuntos, posições e argumentos.

O modelo define igualmente o universo de movimentos retóricos possíveis de efectuar pelos participantes (Figura 5.2). A coordenação é aqui entendida como o controlo pelo sistema destes movimentos retóricos.

Esta abordagem é fundamentalmente destinada a suportar situações de ajustamento mútuo onde a estrutura da discussão (ou mais genericamente, dos conteúdos) tem maior relevo que a estrutura da actividades (coordenação). Por isso mesmo o modelo IBIS é frequentemente combinado com o modelo de dados hipertexto (Conklin, 1988; Rein e Ellis, 1991) ou associado a sistemas de gestão do conhecimento (Shipman e Marshall, 1999).

Deve finalmente ser referido que há dados experimentais que indicam que muitos utilizadores ignoram a estrutura retórica imposta pelo sistema, de que resultam meros sistemas de troca de mensagens (Shipman e Marshall, 1999).

5.5 Coordenação recíproca, por mecanismos linguísticos

Os mecanismos linguísticos de coordenação baseiam-se numa teoria conhecida por actos de fala (*Speech Acts*; Winograd e Flores, 1986; Winograd, 1988).

Na génese desta teoria encontra-se a observação de que existe uma separação artificial entre os conceitos de acção, interacção e linguagem. Nesta perspectiva, a linguagem deixa de constituir apenas um meio de representação, tornando-se igualmente um meio de expressar intenções, necessidades e actividades. Ou seja, constata-se que as interacções entre indivíduos se materializam fundamentalmente através da linguagem.

À comunicação através de linguagem chama-se conversação. A teoria de actos de fala não só associa contexto e lógica à conversação, o que permite que os indivíduos interpretem e caracterizem as mensagens como falsas ou verdadeiras, como também lhe associa intenção.

A intenção introduz um compromisso entre os intervenientes na conversação. De uma intenção resultam expectativas – de que o compromisso seja cumprido – e actividades necessárias ao seu cumprimento.

Um acto de fala que cumpre estas premissas pode ser caracterizado quanto à sua intenção numa de cinco categorias (Winograd e Flores, 1986):

- Assertivo – Intenção de que uma proposição seja verdadeira;
- Directivo – Pedir ao receptor para realizar algo, seja responder a uma pergunta ou executar uma tarefa;
- Comitente – Compromete o emissor a realizar algo;
- Expressivo – Exprime o estado do emissor;
- Declarativo – Associa uma proposição com a realidade.

Ligados em particular à questão da coordenação encontramos os actos de fala directivos (*request*) e comitentes (*offer*). São estes dois actos de fala que permitem interligar e coordenar as actividades individuais realizadas pelos participantes de um grupo, pois suportam dois aspectos chave: o pedido e o compromisso na realização de actividades.

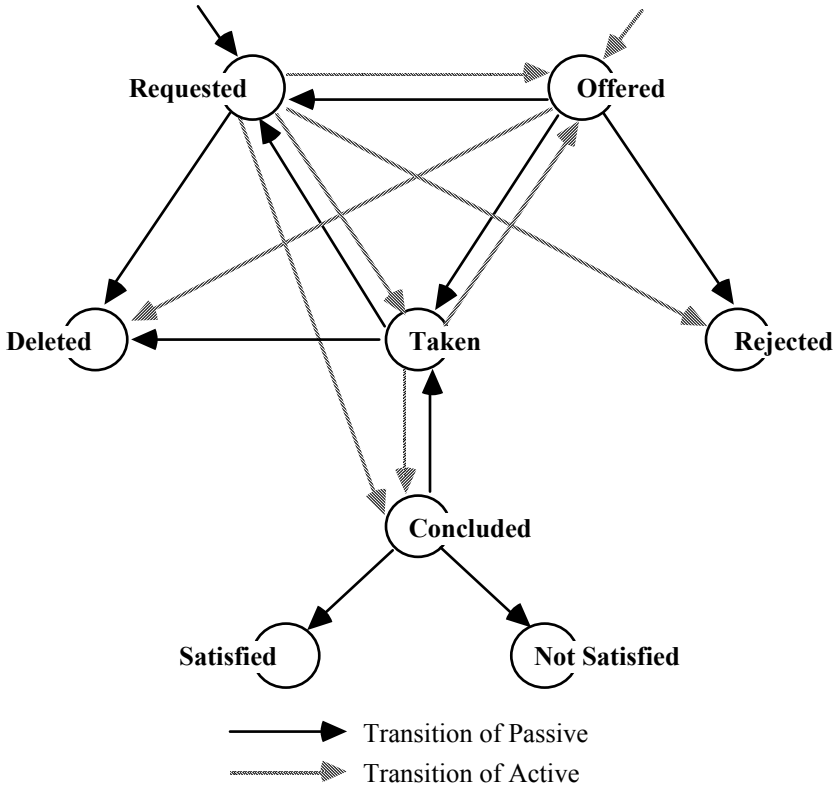


Figura 5.3: Padrão do tipo "commitment to do" (Fonte: Agostini et al., 1994)

A partir dos actos de fala podem então ser estabelecidos diversos padrões de coordenação. Por exemplo, definindo que um "pedido" pode ser "aceite", "rejeitado", "modificado" ou "negociado" (Winograd e Flores, 1986; Rodden e Blair, 1991). Apresentam-se aqui dois exemplos relevantes: um padrão que descreve o compromisso de realizar determinada tarefa (*commitment to do;* na Figura 5.3) e outro que descreve o compromisso de assumir determinado papel (*commitment to be;* na Figura 5.4).

É a partir deste conceito de linguagem como meio de expressar acções que foram desenvolvidos diversos sistemas cooperativos como o The Coordinator, Conversation Builder e UTUCS descritos mais adiante.

A característica fundamental desta aproximação é que ela torna explícita as estruturas complexas de linguagem que habitualmente se encontram implícitas nas interacções face-a-face entre indivíduos.

Os defensores do modelo de actos de fala reclamam que a vantagem de tornar as estruturas de linguagem explícitas se deve a que esta aproximação fornece bases teóricas para interpretar as interacções entre indivíduos,

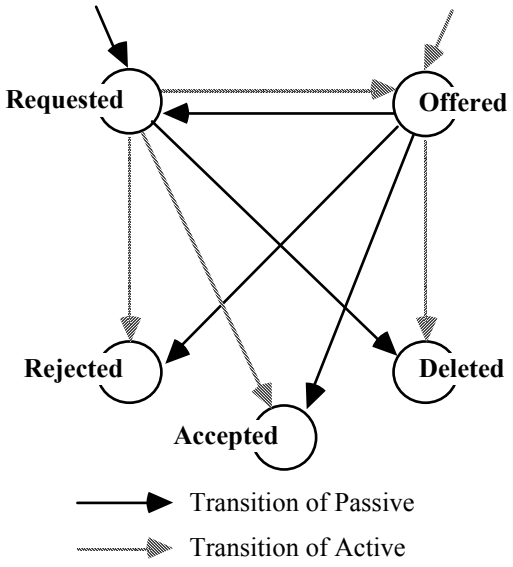


Figura 5.4: Padrão do tipo "commitment to be" (Fonte: Agostini et al., 1994)

exactamente nas situações em que a sua interpretação se torna mais difícil: quando os indivíduos não se encontram face-a-face e utilizam meios computacionais para interactuar.

Os detractores deste modelo, no entanto, consideram que a sua utilização dá origem a sistemas que exercem demasiado controlo sobre os participantes. Sistemas como o Conversation Builder e UTUCS procuram compromissos nestas duas perspectivas que demonstrem que é possível associar as vantagens do modelo de actos de fala com aproximações que exercem menor controlo sobre os participantes.

5.6 Exemplos

The Coordinator

O sistema The Coordinator (Flores et al., 1988) providencia meios computacionais para interligar indivíduos que se coordenam através de actos de fala.

O sistema utiliza o correio electrónico para trocar actos de fala. As mensagens trocadas entre participantes são semelhantes às mensagens semi-formais: cada mensagem contém um campo formal que a classifica no universos de expressões permitidas pelo modelo (por exemplo, "pedido", "aceitação", "rejeição", "modificação", "negociação"). Para cada mensagem

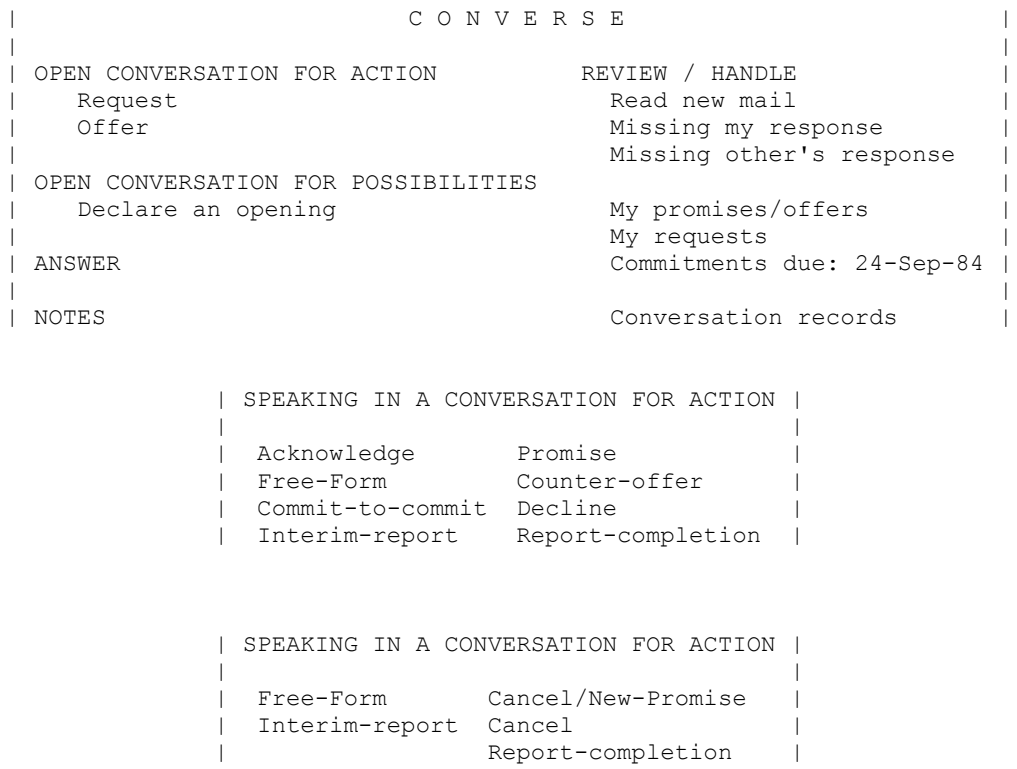


Figura 5.5: Alguns ecrãs do The Coordinator mostrando o controlo efectuado pelo sistema (Fonte: Winograd, 1988)

recebida, os utilizadores do sistema visualizam o seu campo formal e apenas podem gerar mensagens permitidas pelos padrões estabelecidos.

O The Coordinator suporta a geração, armazenamento, pesquisa e visualização de múltiplas conversações (Figura 5.5). O sistema faz uma gestão temporal das conversações dos utilizadores, assinalando a necessidade de prosseguir com as conversações não terminadas.

Conversation Builder

Tal como o The Coordinator, o Conversation Builder (Kaplan et al., 1991, 1992) baseia-se igualmente em conceitos linguísticos para coordenar os seus utilizadores. Um conceito novo, comparativamente ao The Coordinator, introduzido por este sistema é o de protocolo. O conceito de protocolo dá maior relevo aos padrões recorrentes que emergem nas conversações entre indivíduos. Estes padrões (protocolos) são materializados por um conjunto de regras de conversação, definidas e exercitadas ao longo do tempo pelos indivíduos, que definem sequências de actos de fala e a que naturalmente correspondem sequências de actividades.

O Conversation Builder permite que os programadores desenvolvam protocolos específicos para a semântica de cada aplicação. Estes protocolos são especificados na linguagem de programação LISP.

A arquitectura do sistema Conversation Builder é constituída por: (1) uma máquina dedicada a gerir a execução dos protocolos; (2) diversos sub-sistemas de suporte à interface pessoa-máquina, integrando igualmente a visualização do estado de cada protocolo; (3) um sub-sistema de interligação, não só permitindo a comunicação entre os nós do sistema mas também a gestão dos diversos sub-sistemas.

UTUCS/Milano

O sistema UTUCS (Figura 5.6; Agostini et al., 1994; Michelis, 1994), resulta da observação de diversos problemas relacionados com o modelo de actos de fala:

- O modelo de actos de fala assume uma relação directa e única entre conversação e compromisso;

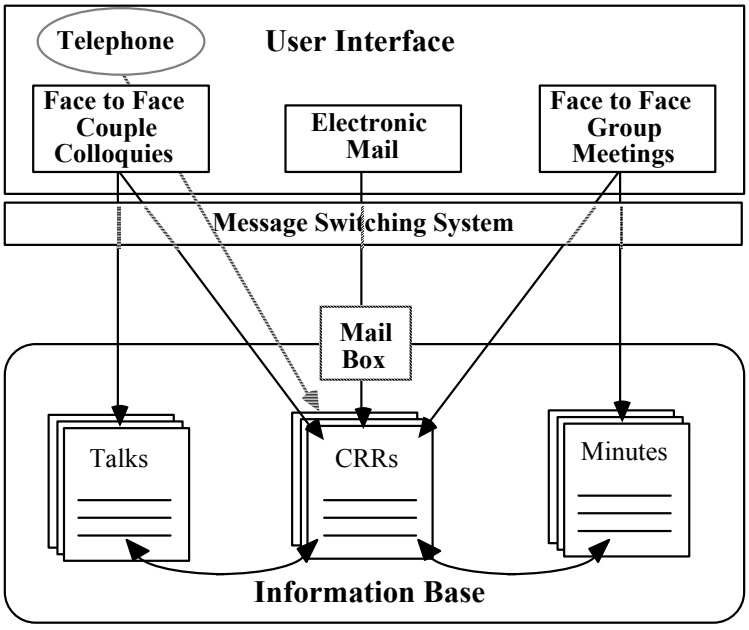
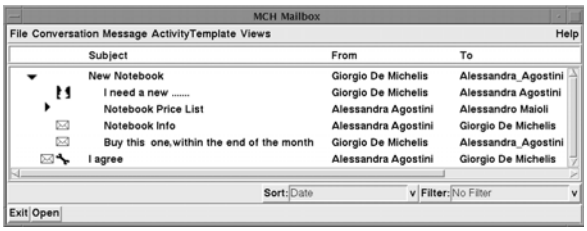


Figura 5.6: A arquitectura do sistema UTUCS integra diversos meios de comunicação, audio video e correio electrónico (Fonte: Agostini et al., 1994)



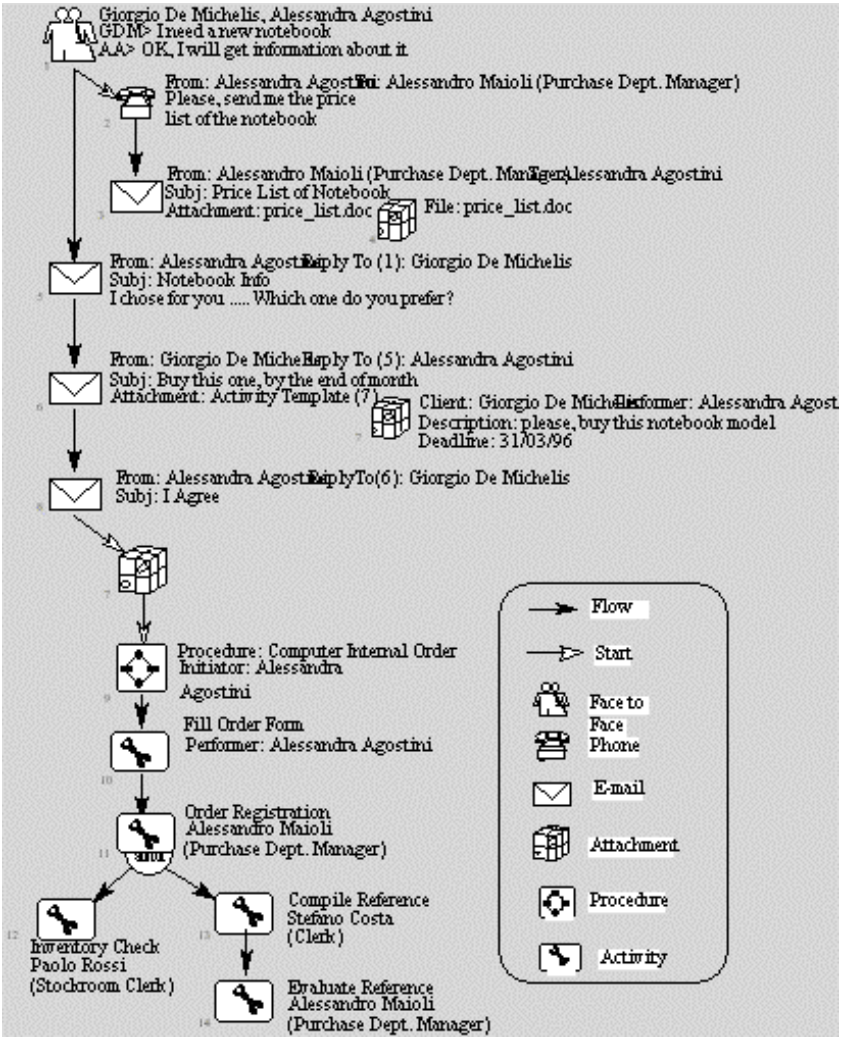


Figura 5.7: Sistema Milano: Fundamentalmente consiste numa pasta que regista informação sobre as conversações e os compromissos assumidos pelos participantes (Fonte: Agostini et al., 1997)

- A utilização do modelo de actos de fala torna-se base de suporte de sistemas normativos, que forçam o compromisso, controlam e disciplinam as interacções;
- O modelo não permite refinamentos adequados a grupos particulares de indivíduos.

Em resumo, o modelo de actos de fala é marcadamente destinado a modelar protocolos normativos de coordenação entre os indivíduos.

Ora uma perspectiva mais abrangente das actividades em grupo diz-nos que as interacções não se constroem apenas a partir de protocolos normativos mas englobam igualmente protocolos casuais (designados por conversações). Desta ideia integradora nasce um novo modelo, denominado Modelo de Conversação de Milão (MCM), que, por sua vez, está na origem do sistema UTUCS e posteriormente do sistema Milano (Figura 5.7).

Segundo o modelo MCM, e contrariamente ao modelo de actos de fala, a coordenação entre os indivíduos caracteriza-se, num primeiro plano, por não explicitar qualquer protocolo normativo, apesar de ele poder existir. Ou seja, as interacções, na faceta de conversações, exibem a natureza informal e comunicativa das relações humanas.

Num segundo plano surge o conceito de compromisso entre indivíduos, que se rege por protocolos normativos que seguem as regras identificadas no modelo de actos de fala. O compromisso distingue-se da conversação por explicitar e formalizar os protocolos seguidos pelos indivíduos. Ou seja, as interacções, na faceta de compromissos, exibem a natureza formal e metódica das relações humanas.

A coordenação entre os indivíduos é definida pela interligação entre conversações (casuais) e compromissos (normativos), sendo o contexto de um processo cooperativo caracterizado pelo historial de conversações e compromissos assumidos pelos seus intervenientes.

O sistema Milano (agostini et al., 1997) suporta a geração de conversações e compromissos e permite analisar o historial dos processos cooperativos. As conversações podem ser multimédia (telefone, vídeo e correio electrónico).

ProMinanD

Ilustrativo da utilização dos mecanismos sequenciais formais de coordenação é o sistema ProMinanD (Karbe e Ramsperger, 1990). Este

sistema modela a interacção entre indivíduos através de pastas de informação que circulam num escritório electrónico. A chegada de uma pasta a um utilizador desencadeia uma actividade individual que, quando terminada, resulta na deslocação da pasta em direcção a outro utilizador, definindo-se deste modo um fluxo de trabalho em grupo.

Neste tipo de mecanismo, a coordenação é efectuada pelo sistema de encaminhamento de pastas.

No sistema ProMinanD, uma pasta é constituída por uma parte descritiva, contendo a identificação do documento, estado do fluxo, ligações a outras pastas, etc; e outra parte contendo informação diversa. O sistema suporta o encaminhamento de pastas e inclui mecanismos de processamento de excepções: (1) selecção de fluxos alternativos, na impossibilidade de entregar uma pasta a determinado indivíduo; (2) mover o documento para trás, quando um indivíduo assinala que a pasta lhe foi incorrectamente enviada; (3) inserção dinâmica de indivíduos no fluxo do documento; e (4) pesquisa automática de receptores alternativos, utilizando para isso definições dos papéis de cada indivíduo.

6 Espaço público

Em qualquer sistema ou aplicação mono-utilizador a função da interface pessoa-máquina é actuar como mediadora entre utilizador e sistema computacional. Originalmente, a interface pessoa-máquina consistia na gestão das entradas/saídas do sistema.

Nos sistemas computacionais modernos, constituídos por estações de trabalho com ecrã, rato e teclado, o grau de complexidade da interface pessoa-máquina é bastante mais elevado, considerando o suporte à manipulação de objectos gráficos que se apresentam ao utilizador.

Um sistema onde interactuam múltiplos utilizadores alarga ainda mais as competências da interface pessoa-máquina. Este alargamento realiza-se em particular ao nível dos objectos gráficos partilhados pelos utilizadores do sistema, definindo o conceito de espaço público.

O conceito de espaço público surgiu originalmente no sistema NLS/augment (Engelbart, 1988a, 1998b), desenvolvido na Universidade de Stanford em 1968. Alguns dos ensaios realizados pela equipa de projecto envolveram a utilização de seis ecrãs vectoriais (desenvolvidos especialmente para esse projecto) que mostravam simultaneamente o mesmo conteúdo aos utilizadores do sistema (Figura 6.1).

Figura 6.1: Demonstração do sistema NLS/Augment

Pode assim definir-se um espaço público como sendo um conjunto de ecrãs de um sistema que mostram simultaneamente a mesma informação a diversos utilizadores. Esta definição é no entanto bastante restritiva, condicionando muito severamente a arquitectura do sistema por causa do requisito de simultaneidade. Por exemplo, no sistema NLS/augment o espaço público é centralizado numa sala e constituído por ecrãs vectoriais ligados fisicamente entre si (em paralelo). Só assim o sistema é capaz de efectivamente mostrar a mesma informação a todos os utilizadores em qualquer instante.

Uma definição mais flexível caracteriza um espaço público como sendo um conjunto de secções dos ecrãs de um sistema em que cada um dos utilizadores observa a mesma informação. (Sarin e Greif, 1985). Esta definição transforma o espaço físico num espaço virtual, caracterizado por janelas gráficas (Figura 6.2); e dá relevo à coerência da informação perceptível pelos utilizadores, ao invés de focar na simultaneidade da sua apresentação. A partir desta definição já é possível conceber um sistema remoto, constituído por diversos nós que comunicam através de redes de dados, sendo no entanto necessário garantir que o tempo de resposta permita algum grau de coerência na observação da informação.

Finalmente, uma outra definição ainda mais flexível define um espaço

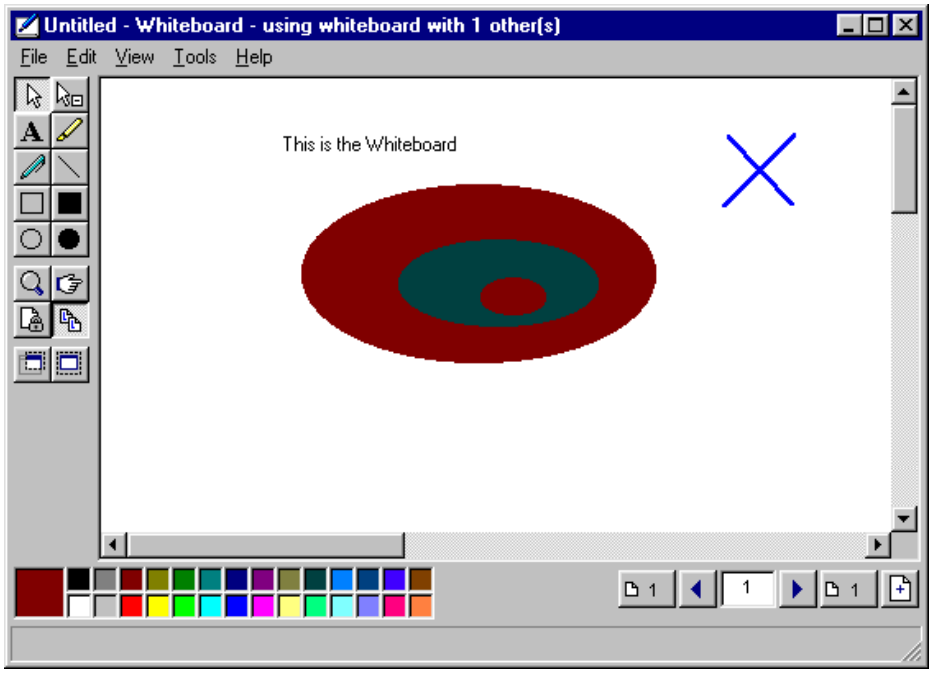


Figura 6.2: Um típico espaço público, apresentado numa janela do ecrã de cada utilizador

Figura 6.3: Um espaço público virtual, englobando um ecrã de grandes dimensões e diversos ecrãs de computadores pessoais (Fonte: http://www.darmstadt.gmd.de)

público como sendo um espaço gráfico, partilhado e actualizado dinamicamente a partir de um ou mais centros de controlo (Kamel, 1993). Esta definição sugere uma interpretação ainda mais fraca quanto à própria coerência da informação que é fornecida aos utilizadores, ao considerar apenas que o espaço público é partilhado e actualizado dinamicamente. Esta definição é particularmente útil para permitir total liberdade quanto ao tempo de resposta na comunicação de mensagens.

Em resumo, pode-se concluir que um espaço público é caracterizado pela coerência da informação que apresenta aos seus utilizadores, sendo no entanto possível definir diversos graus de coerência. Os modelos que se apresentam de seguida permitem caracterizar os espaços públicos segundo esse grau de coerência.

6.1 WYSIWIS estrito

O modelo WYSIWIS (*What You See Is What I See*) estrito (Stefik et al., 1987) assegura que o espaço público apresenta estritamente a mesma informação a todos os utilizadores. Neste conceito, engloba-se a seguite informação:

- O conteúdo dos objectos gráficos presentes no espaço público.

Figura 6.4: Modelo WYSIWIS estrito: Note-se que o menu que surge no ecrã de cima surge igualmente nos restantes ecrãs (Fonte: www.smarttech.com)

- Todos os atributos desses objectos, como posição, tamanho, cor, etc.
- Toda a informação de retorno que é fornecida aos utilizadores quando operam sobre os objectos; quando por exemplo movimentam objectos (Figura 6.4).

Uma característica importante desta definição estrita é que ela atribui ao espaço público um forte sentido de convergência, permitindo que os utilizadores recorram a referências contextuais, empregando termos como "este objecto," "aqui" ou "agora" (Ellis et al., 1991).

Devem no entanto notar-se alguns problemas no modelo WYSIWIS estrito. A experiência com este modelo mostrou a necessidade de suportar uma manipulação de objectos mais flexível (Dewan, 1991), pois o trabalho em grupo é uma conjugação de actividades fortemente cooperativas com outras mais individualizadas.

Por outro lado, em sistemas tempo-igual/local-diferente, este modelo apresenta requisitos de largura de banda que podem não ser facilmente suportados.[1]

[1] Note-se que o espaço público contém, por definição, informação que é partilhada por vários utilizadores. Logo, ao nível da interface pessoa-máquina, levantam-se as questões de arquitectura, comunicação e concorrência discutidas anteriormente.

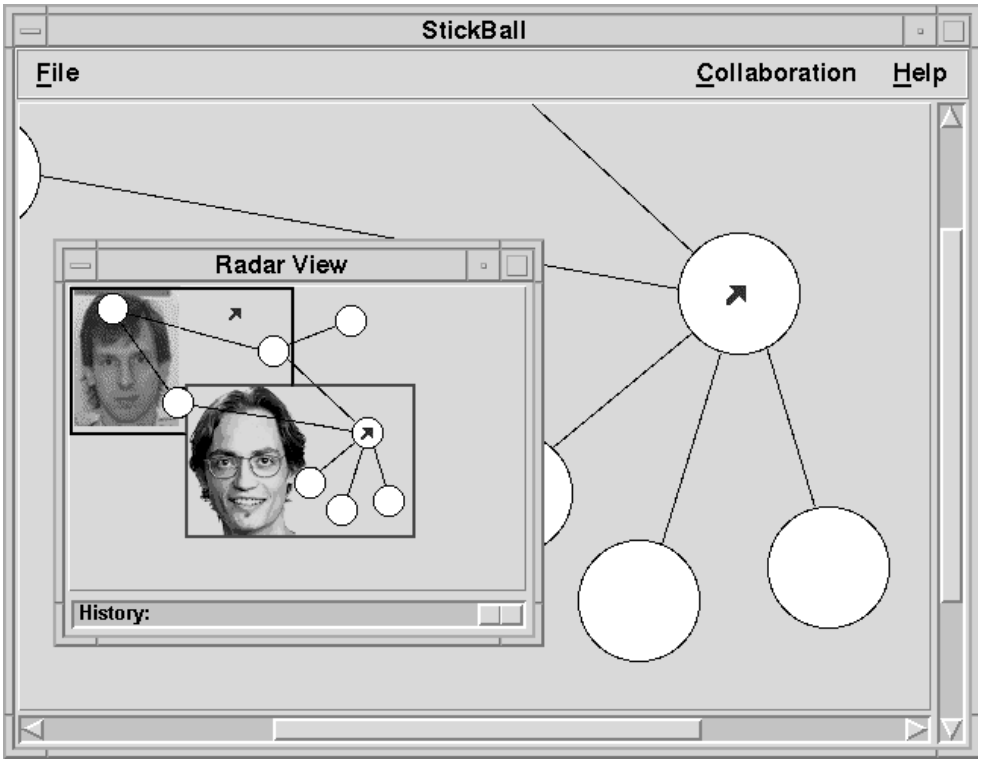


Figura 6.5: Modelo WYSIWIMS: Dois utilizadores partilham o mesmo espaço público mas com vistas diferentes sobre o espaço (Fonte: Gutwin et al., 1996)

O modelo WYSIWIS estrito é, portanto, de aplicação muito particular. Esta constatação deu origem aos modelos WYSIWIMS, WYGIWIG e WYSIWIS relaxado, que procedem ao relaxamento da coerência do espaço público em múltiplas dimensões.

6.2 WYSIWIMS

O modelo WYSIWIMS (*What You See Is What I May See*) relaxa espacialmente a coerência dos dados de modo a permitir que cada utilizador observe secções diferentes do espaço público (Figura 6.5).

Este relaxamento realiza-se através de múltiplas vistas (*viewports*) sobre um espaço público virtual de grande dimensão (Greenberg, 1990). Cada utilizador pode movimentar livremente a sua vista sobre o espaço público sem influenciar as actividades dos restantes utilizadores do sistema.

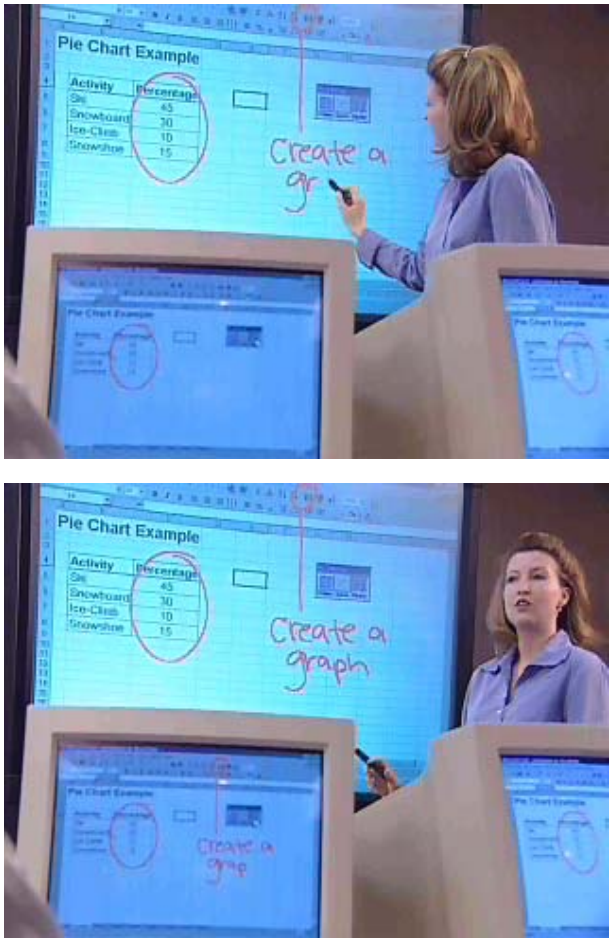


Figura 6.6: Ilustração do modelo WYGIWIG: Note-se, no ecrã de cima, que num determinado instante temporal a informação é incoerente (Fonte: www.smarttech.com)

O relaxamento espacial inviabiliza a utilização de referências contextuais relacionadas com o posicionamento dos objectos.

6.3 WYGIWIG

O modelo WYGIWIG (*What You Get Is What I Get*) permite o relaxamento temporal da coerência dos dados e decorre da análise de dois problemas de comunicação já discutidos anteriormente.

Em primeiro lugar, tendo o espaço público informação partilhada pelos utilizadores, sempre que ocorre alguma alteração num nó torna-se necessário difundir mensagens para os restantes nós do sistema para repor a coerência do espaço público. No entanto, a coerência só será atingida após o tempo de notificação associado à transmissão das mensagens (Figura 6.6).

Como podem existir diversos utilizadores concorrentemente a utilizar o sistema, essa coerência pode realmente nunca existir.

Em segundo lugar, o espaço público lida com eventos de baixa granularidade, por exemplo, o arrastamento de um objecto no espaço público decorre de uma sequência de eventos gerados pelo rato. Ora manter a coerência desse movimento no espaço público exige uma elevada largura de banda que nem sempre pode ser suportada pelo sistema.

O modelo WYGIWIG garante que as actualizações dos objectos presentes no espaço público se repercutem a todos os utilizadores, mas após um intervalo de tempo variável e não determinado, pelo que a coerência do espaço não deve ser assumida. O modelo WYGIWIG também não garante a coerência de eventos de baixa granularidade efectuados sobre os objectos do espaço público.

Por exemplo, Cook et al. (1991) restringem a coerência às posições iniciais e finais dos objectos movimentados no espaço público, não considerando as posições intermédias.

Outro exemplo interessante: o GroupDesign (Beaudouin-Lafon e Karsenty, 1992) permite igualmente que um utilizador mova um objecto sem que esse movimento seja repercutido aos restantes utilizadores. Apenas a localização final é transmitida. No entanto, para preservar o contexto, o sistema apresenta a cada utilizador uma animação gráfica, representativa de que o objecto foi movido.

6.4 WYSIWIS relaxado – Outras dimensões

Para além do espaço e tempo, a coerência do espaço público pode ainda ser relaxada nas seguintes dimensões:

- População – Define um subconjunto de utilizadores que interactuam com o espaço público em modo estrito. Os restantes utilizadores observam versões do espaço público. Por exemplo, o GroupDesign (Beaudouin-Lafon e Karsenty, 1992) permite que um utilizador suspenda a recepção de actualizações de objectos no espaço público; o rIBIS (Rein e Ellis, 1991) permite que cada utilizador se associe/dissocie do modo estrito.
- Atributos – Define um subconjunto de atributos dos objectos para os quais a coerência não se aplica em modo estrito. Este tipo de relaxamento permite personalizar alguns atributos do espaço público e evitar guerras de configuração (*scroll wars*) entre os utilizadores (Stefik et al., 1987). Por exemplo, o GROVE (Ellis et al., 1991)

atribui cores diferentes aos objectos para revelar aos utilizadores as suas permissões de leitura/escrita.

- Semântica – Uma aproximação clássica dos sistema de interface pessoa-máquina consiste em separar os dados da aplicação dos dados unicamente relacionados com a interface. O relaxamento de atributos do espaço público pode ser efectuado de acordo com estes dois tipos de dados.

  Por exemplo, o 4D (Antunes et al., 1992, 1993) define objectos semânticos, de apresentação e de diálogo. Os objectos de apresentação servem de mediadores entre utilizadores e aplicação enquanto que os objectos de diálogo, que se interpõem entre objectos de apresentação e objectos semânticos, configuram a interacção com os utilizadores.

  O relaxamento do espaço público efectua-se diferenciando o modelo de coerência para cada um destes tipos de objectos.

6.5 Relação entre espaço público e espaço privado

A maioria das actividades em grupo exige que o sistema cooperativo complemente os espaços públicos com espaços privados, i.e. espaços onde as interacções com o sistema se circunscrevem a um único utilizador.

As razões que contribuem para esta necessidade incluem:

- Permitir que os utilizadores particularizem os seus interesses e objectivos (Sarin e Greif, 1985; Dourish e Bellotti, 1992);
- Maior eficiência na execução de tarefas que podem ser decompostas em múltiplas sub-actividades paralelas (Stefik at al., 1987);
- E necessidade de privacidade durante as sessões cooperativas, por exemplo, para evitar embaraço público quando o utilizador não se encontra familiarizado com o sistema (Elwart-Keys et al., 1990; Beaudouin-Lafon e Karsenty, 1992).

Todavia, a introdução de espaços privados apresenta alguns problemas para o sistema cooperativo. Nomeadamente, a comunicação contextual entre os utilizadores do sistema torna-se mais complexa, pois os utilizadores inadvertidamente tentam referenciar informação que se encontra nos seus espaços privados.

Outro problema é que os utilizadores, por vezes, sentem a necessidade de observar a informação presente nos espaços privados de outros utilizadores, o que colide com a funcionalidade destes espaços. Os resultados de utilização de alguns sistemas revelam ainda que a possibilidade de os

utilizadores efectuarem actividades prolongadas nos seus espaços privados causa igualmente dificuldades no retorno a actividades conjuntas, por perda de contexto partilhado (Miller et al., 1992).

As soluções para os problemas criados pelos espaços privados passam por uma interligação suave com os espaços públicos, resultando na definição de espaços semi-privados (Brothers et al., 1990; Dewan, 1991).

Os espaços semi-privados proporcionam a identificação dos objectos nele presentes assim como a eventual observação das acções exercidas sobre esses objectos, sem no entanto garantir a partilha total.

Apresentam-se algumas alternativas para a concretização de espaços semi-privados:

- Associar informação privada ao espaço público – Definindo campos privados nos objectos presentes no espaço público. Por exemplo, o DOLPHIN (Streitz et al., 1994) insere referências no espaço público a objectos presentes nos espaços privados dos utilizadores.
- Aproximação matricial – O espaço público é fragmentado em matriz. Um utilizador observa os objectos presentes em todas as células mas apenas pode manipular uma célula activa. O controlo de acesso é efectuado ao nível da célula. Esta aproximação parte do princípio que cada célula agrupa dados que tipicamente devem ser manipulados por um único utilizador.
- Aproximação hipertexto – A organização dos dados em hipertexto permite seleccionar o modelo de espaço público em cada nó.

  A selecção do modelo pode ser automatizada. Por exemplo, o sistema SEPIA (Haake e Haake, 1992) actualiza o modelo de cada nó de acordo com as acções dos seus utilizadores: um nó manipulado por um único utilizador comporta-se como um espaço privado; se outros utilizadores visitarem o mesmo nó ele comporta-se igualmente como um espaço privado; se diversos utilizadores pretenderem manipular esse nó, o espaço torna-se público.

- Superfícies transparentes – O espaço semi-privado é construído a partir de um conjunto de superfícies sobrepostas e transparentes. Um utilizador apenas pode manipular os objectos presentes nas superfícies por ele geradas, pelo que não existe concorrência física entre utilizadores. No entanto, como a informação é partilhada visualmente, pois um utilizador observa os objectos gerados pelos restantes utilizadores, é possível gerar objectos sobrepostos.

Posteriormente, através da negociação entre utilizadores, as sobreposições podem ser eliminadas, o que efectivamente resulta na partilha dos objectos presentes no espaço semi-privado (Roseman e Greenberg, 1992).

- Superfícies com filtros – Consiste igualmente na sobreposição de superfícies transparentes, mas onde os objectos presentes em camadas inferiores vão perdendo progressivamente a visibilidade (Lu e Mantei, 1991).

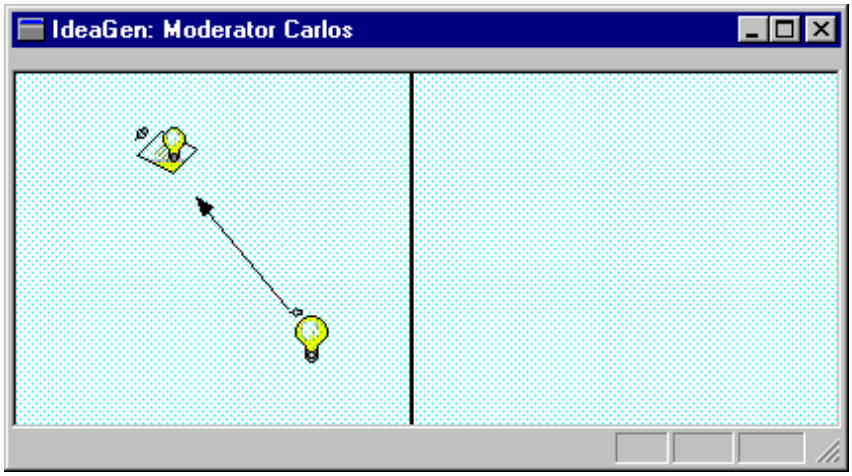


Figura 6.6a: IdeaGen: Espaço privado à esquerda e espaço público à direita

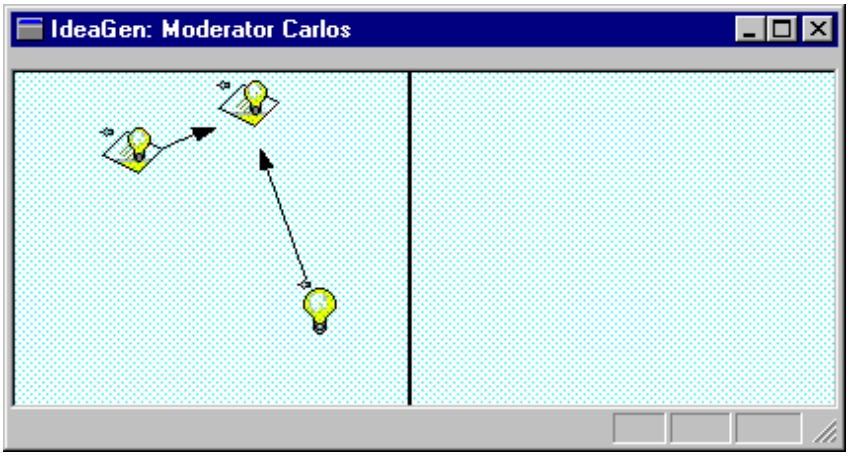


Figura 6.6b: Estruturação de ideias no IdeaGen

6.6 Exemplo

IdeaGen

O sistema IdeaGen combina espaços públicos com espaços privados.

A geração de ideias no IdeaGen é realizada no espaço privado de cada utilizador (Figura 6.6a). O utilizador, no seu espaço privado, é livre de criar e estruturar ideias do modo que entender.

Como foi referido anteriormente, a estruturação das ideias obedece a uma forma hierárquica, no topo da hierarquia existe obrigatoriamente uma "pasta" de ideias e em níveis inferiores podem existir não só ideias mas também "pastas" com ideias (Figura 6.6b).

Após a geração privada das ideias o utilizador tem a possibilidade de tornar a estrutura de ideias visível aos restantes elementos do grupo, bastando para tal arrastá-la para o espaço público. Neste espaço todos os utilizadores podem editar os conteúdos, alterar a estruturação das ideias e as associações entre "pastas" de ideias.

O espaço público do IdeaGen é do tipo WYSIWIS relaxado, sendo permitido aos utilizadores modificar diversos atributos dos objectos no espaço público sem que essas modificações sejam propagadas por todos os

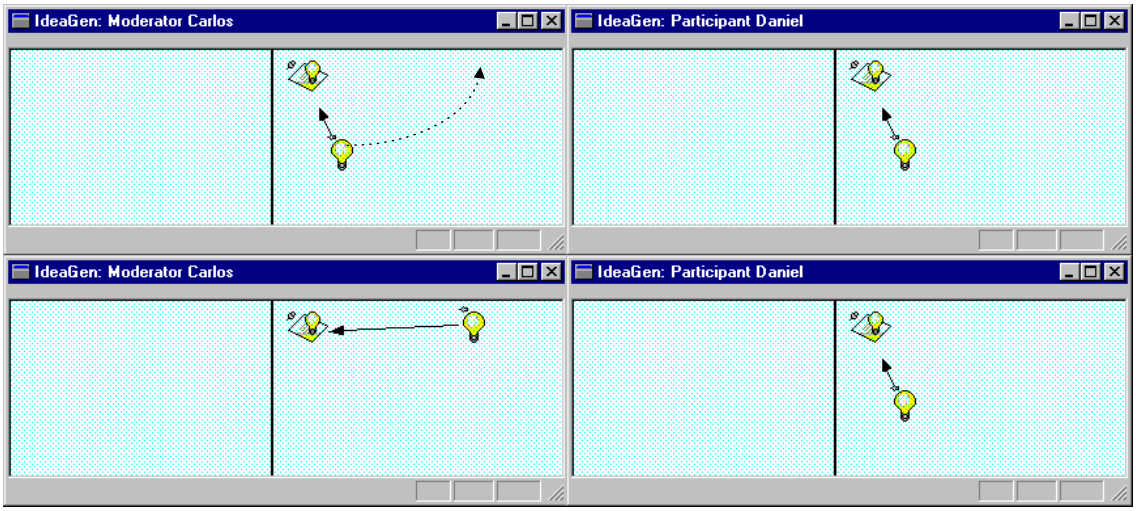


Figura 6.7: Ilustração do relaxamento do espaço público: O utilizador da esquerda move o objecto no espaço público sem influenciar a vista do utilizador da direita

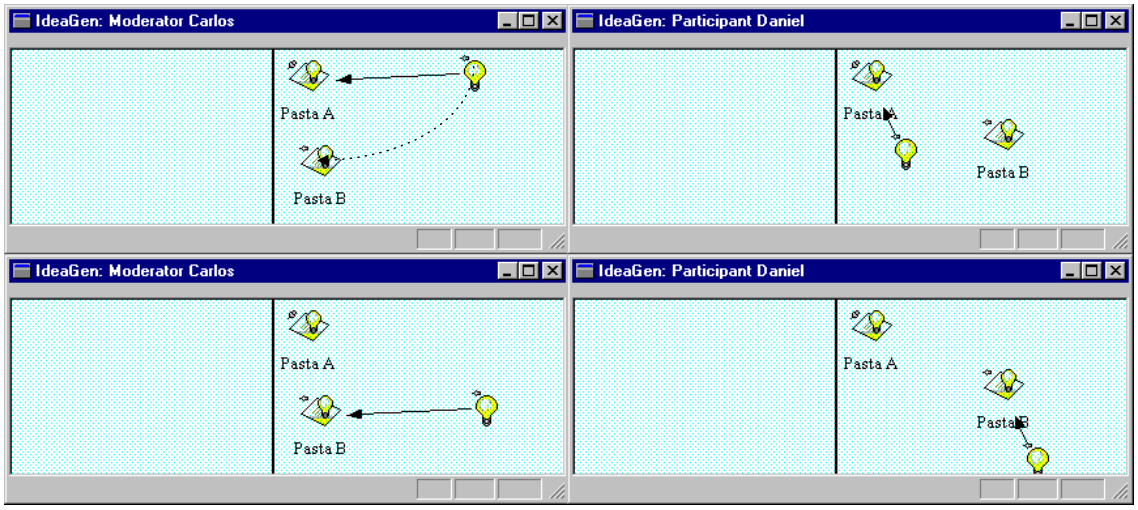


Figura 6.8: As alterações à estrutura dos objectos são geridas em modo estrito: O utilizador da esquerda muda a hierarquia dos objectos e essa modificação é apresentada ao utilizador da direita

utilizadores. Um desses atributos é a localização do objecto no espaço público (Figura 6.7).

Em contraponto, as mudanças na estrutura hierárquica dos objectos são propagadas por todos os utilizadores (Figura 6.8).

7 Monitorização

Num sistema mono-utilizador a interface pessoa-máquina rege-se por um princípio de retro-alimentação simples (*feedback*): o utilizador exerce uma acção sobre a interface, que desencadeia uma operação do sistema, que por sua vez produz uma reacção da interface, que é monitorizada pelo utilizador.

As reacções da interface pessoa-máquina são necessárias para permitir ao utilizador compreender as operações do sistema e reagir ou adaptar o seu plano de acção.

A monitorização é assim uma atitude consciente e indispensável de vigilância, ao contrário da percepção, que é uma atitude inconsciente do indivíduo face ao meio ambiente que o envolve.

Um sistema multi-utilizador deve reger-se pelo mesmo princípio, se bem que a situação se torne mais abrangente: uma operação do sistema, desencadeada por uma acção de um utilizador, origina uma reacção que deve ser monitorizada pelo próprio (*feedback*) mas também pelos restantes utilizadores (*feedtrhough*). Neste caso, as razões justificativas da monitorização de acções exercidas no sistema por outros utilizadores devem-se a que ela estimula cada indivíduo a planear e desencadear as suas próprias acções, evitando a duplicação de trabalho, assim como permite avaliar a disponibilidade e oportunidade para comunicar com outros utilizadores (Berlage e Sohlenkamp, 1999).

Em situações mais complexas, o mecanismo de retro-alimentação necessita de incorporar informação de retorno dos restantes utilizadores do sistema (*feedback* + *feedthrough*). Por exemplo, quando um utilizador dirige os outros utilizadores para determinada área do espaço público e necessita de evidências de que os utilizadores se deslocaram para essa área (Gutwin e Greenberg, 1999).

Dados experimentais sobre a utilização do sistemas BSCW (Appelt, 2001) indicam que as funcionalidades do sistema relacionadas com monitorização correspondem à segunda categoria de funcionalidades mais utilizadas (13.4%), sendo apenas suplantada pela criação de informação (24.7%).

A monitorização destina-se a responder às seguintes questões (Berlage e Sohlenkamp, 1999):

- Quem está a fazer algo (origem);

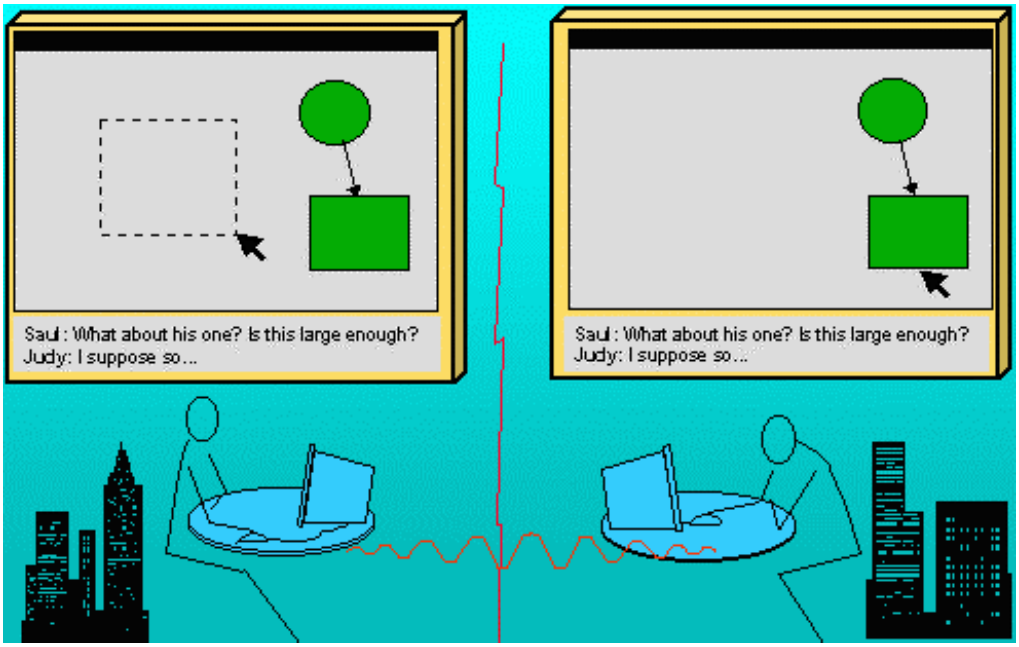


Figura 7.1: O mecanismo de Teleponteiro permite focar o diálogo numa área do espaço público (Fonte: Greenberg e Gutwin, 1998)

- Onde é que o está a fazer (localização);
- O que está a fazer (acção);
- Como é que o está a fazer (invocação);
- Porque é que o está a fazer (motivação);
- Quando é que o fez (tempo).

Por definição, a monitorização está associada a sistemas cooperativos com comunicação síncrona.

Iremos distinguir dois tipos fundamentais de monitorização: convergente e divergente.

7.1 Monitorização convergente

Afirma-se que um sistema suporta monitorização convergente (*tightly coupling*; Rein e Ellis, 1991) quando permite que a atenção que os utilizadores prestam ao sistema seja simultânea e focada no mesmo propósito.

A monitorização convergente complementa e reforça a noção de espaço público discutida anteriormente: se o espaço público é um espaço gráfico, partilhado e actualizado dinamicamente, então os mecanismos de monitorização permitem a um utilizador compreender quais são as acções que os restantes utilizadores se encontram a executar no espaço público.

Figura 7.2: Integração de espaços públicos com imagens dos utilizadores (Fonte: Ishii et al., 1994)

Um exemplo concreto de monitorização convergente suportada pelo espaço público é proporcionado pelo mecanismo gráfico conhecido por tele-ponteiro (*telepointer*; Crowley et al., 1990).

O tele-ponteiro (Figura 7.1) é um objecto gráfico que pode ser movido por um utilizador para chamar a atenção dos restantes utilizadores do sistema para determinada área do ecrã. Os movimentos do tele-ponteiro são observados por todos os utilizadores, o que permite transmitir informação gestual.

Uma das formas melhor conseguidas de implementar a monitorização convergente resulta da integração de um espaço público com superfícies que apresentam imagens reais dos utilizadores, possibilitando o acompanhamento visual das acções que estes realizam sobre os objectos presentes no espaço público (Figura 7.2; Ishii et al., 1994).

7.2 Monitorização divergente

Tome-se como exemplo de monitorização divergente um sistema cooperativo que apenas suporta concorrência, recorrendo ao mecanismo de trinco para controlar o acesso aos dados.

Neste sistema, um utilizador que tente aceder (acção) a um objecto que esteja a ser manipulado por outro utilizador desencadeia uma resposta negativa (reacção) do sistema, a informar da impossibilidade de acesso e, eventualmente, da causa que levou a essa resposta.

Este tipo de monitorização, designada divergente (*loosely coupling*), torna o contexto partilhado perceptível para o utilizador mas, ao contrário da monitorização convergente, não considera que a atenção dos utilizadores seja simultânea e focada no mesmo propósito (Rein e Ellis, 1991).

A noção de monitorização divergente dá origem aos espaços semi-privados dos utilizadores.

7.3 Mecanismos de monitorização

A comparação entre monitorização convergente e divergente revela dois extremos de funcionalidade da interface pessoa-máquina de um sistema cooperativo. No primeiro caso, o sistema fornece aos utilizadores grande quantidade de informação e pressupõe a sua atenção permanente, o que pode tornar-se demasiado intrusivo (Ellis et al., 1991). No segundo caso, a situação é inversa, pois o sistema apenas reage após um estímulo do utilizador.

No entanto, a interface pessoa-máquina pode concretizar graus intermédios de monitorização, capazes de aumentar a vigilância dos utilizadores sem no entanto exigirem uma atenção permanente (Dourish e Bellotti, 1992).

O nível de sofisticação da interface pessoa-máquina pode assim variar, dependendo dos mecanismos que forem disponibilizados pelo sistema cooperativo.

Do ponto de vista do design de sistemas cooperativos, consideram-se três abordagens distintas para a concretização de mecanismos de monitorização (Erikson e Kellogg, 2000):

- Realista – O sistema tenta reproduzir a informação de retorno que as pessoas transmitem nos seus encontros face-a-face. Por exemplo, os sistemas de tele-conferência utilizam esta abordagem.
- Mimética – O sistema tenta criar novas representações para o espaço físico, definindo espaços virtuais bi ou tridimensionais. Os sistemas de MUDs (Multi-User Domain) seguem esta abordagem ao definirem salas virtuais por onde os utilizadores se podem deslocar.
- Abstracta – O sistema fornece informação que não está relacionada com o espaço físico. Por exemplo, os sistemas de *Chat* seguem essa

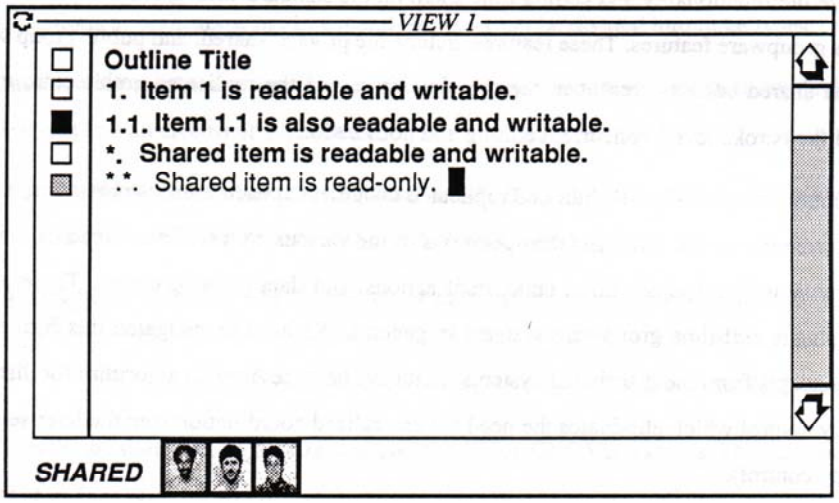


Figura 7.3: Exemplo da identificação de domínios de interesse: Os rectângulos à esquerda indicam quem está a editar um parágrafo (Fonte: Ellis et al., 1991)

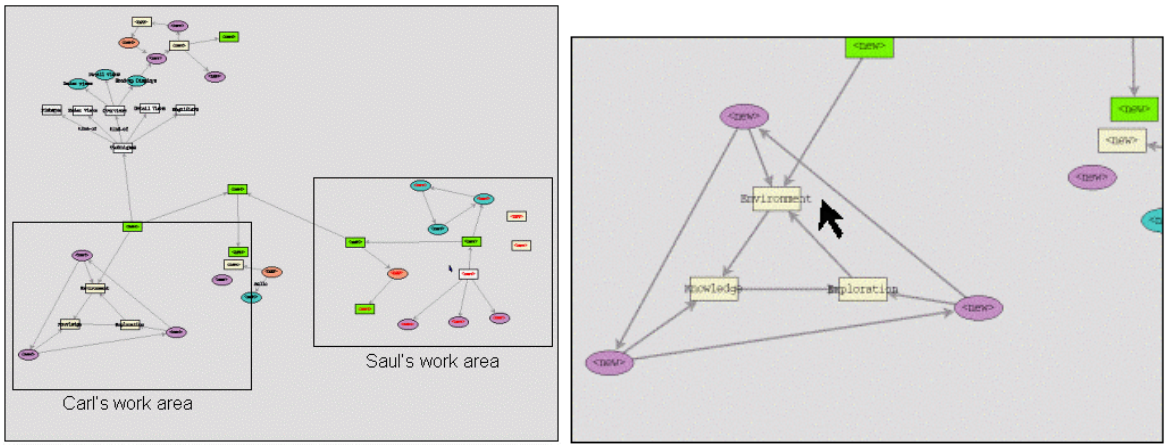


Figura 7.4: Mecanismo de *teleport*: À esquerda o utilizador selecciona uma vista e à direita observa os movimentos do detentor da vista (Fonte: Gutwin e Greenberg, 1998)

abordagem, pois estruturam texto (e outros elementos gráficos) que não reproduz nem o texto escrito (prosa) nem a comunicação oral.

Apresentam-se em seguida diversos exemplos concretos de mecanismos de monitorização divergente utilizados por sistemas cooperativos. Esses mecanismos encontram-se organizados segundo o fim a que se destinam.

Monitorização do espaço de trabalho (*workspace awareness*)

Lida com a informação necessária à manutenção do espaço público, nomeadamente onde estão os diversos participantes e o que estão a fazer:

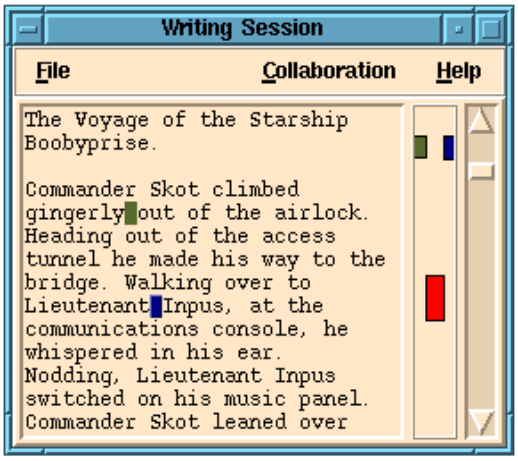


Figura 7.5: Barras de escorregamento multi-utilizador (Fonte: Roseman e Greenberg, 1996)

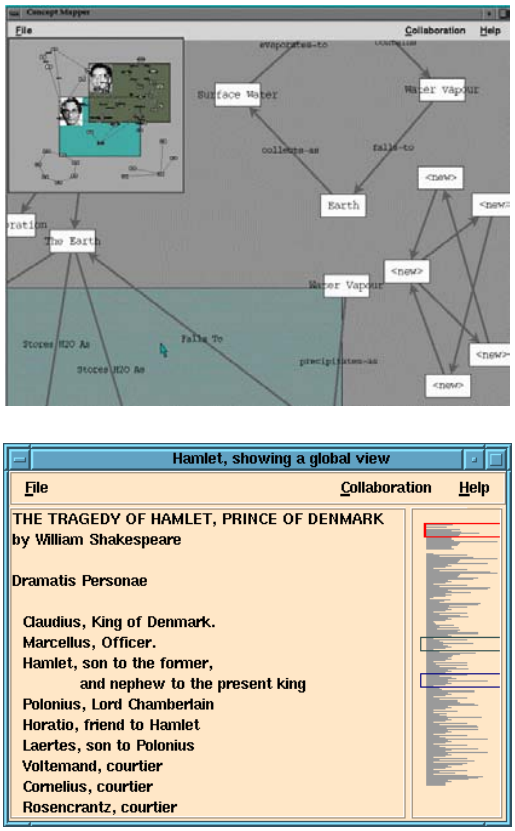


Figura 7.6: Duas vistas de radar (Fonte: Roseman e Greenberg, 1996)

- Identificação dos domínios de interesse – Utilização de cores distintas para identificar os objectos manipulados por cada utilizador no espaço público (Figura 7.3).
- *Teleports* – Um *teleport* é uma vista sobreposta à de outro utilizador, permitindo assim observar as suas movimentações no espaço de trabalho (Figura 7.4; Beaudouin-Lafon e Karsenty, 1992; Roseman e Greenberg, 1996).
- Barras de escorregamento multi-utilizador – As barras de escorregamento coloridas associadas ao espaço de trabalho mostram o posicionamento relativo das vistas dos utilizadores (Figura 7.5; Roseman e Greenberg, 1996).
- Vistas de radar – São vistas condensadas do espaço de trabalho que localizam e mostram os movimentos dos seus diversos utilizadores (Figura 7.6; Roseman e Greenberg, 1996; Gutwin e Greenberg, 1999).
- Nuvens (*gaze awareness*) – Definem áreas bidimensionais ou tridimensionais de interesse dos utilizadores à volta de cada objecto presente no espaço público. Uma métrica, baseada na intersecção entre duas áreas de interesse, serve para caracterizar o grau de interesse de dois utilizadores sobre o mesmo objecto e permite configurar a sua interacção (por exemplo, uma pequena intersecção quer dizer que o utilizador apenas quer observar o objecto e uma grande intersecção quer dizer que o utilizador pretende editar o objecto; Benford, 1993).
- Indicadores de actividade – Identificam no espaço de trabalho a natureza das manipulações de objectos e a identidade de quem as realizou (Dourish e Bellotti, 1992). Essa informação pode ter um limite temporal ou mesmo ser esbatida ao longo do tempo (Ellis et al., 1991).
- Historial de actividades – Este mecanismo permite que um utilizador observe a sequência de modificações que foram executadas por outro utilizador sobre um objecto (Beaudouin-Lafon e Karsenty, 1992).

Monitorização do grupo de trabalho

Numa interacção face-a-face o reconhecimento dos participantes é implícito, resultando de sinais visuais e auditivos. Quando a interacção é remota, emerge a dificuldade em reconhecer a identidade dos indivíduos que participam no processo.

As soluções para este problema passam por uma série de mecanismos que fornecem informação geral sobre quem está a utilizar o sistema:

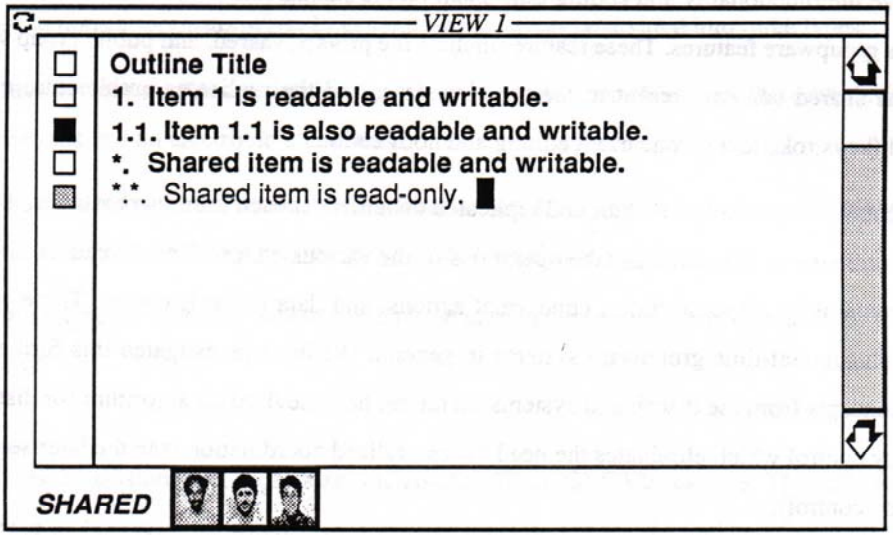


Figura 7.7: Identificação dinâmica de utilizadores: Apresentação das fotografias dos utilizadores (Fonte: Ellis et al., 1991)

Figura 7.8: *Peepholes*: Imagens reais do posto de trabalho de cada utilizador (incluindo mecanismos de garantia da privacidade; Fonte: Greenberg e Kuzuoka, 1999)

- Identificação dinâmica de utilizadores – Este mecanismos apresentam ao utilizador uma janela com os nomes dos restantes utilizadores do espaço de trabalho (Figura 7.7; Dourish e Bellotti, 1992);
- Indicadores de disponibilidade (*peepholes*) – Dão informação sobre a disponibilidade dos utilizadores (Figura 7.8; Greenberg, 1996);
- Indicadores de emoções – utilizam representações gráficas de faces – com olhos, boca, nariz, orelhas e cabelo – que podem ser personalizadas por cada utilizador para expressar emoções: acordo, desacordo, tristeza, irritação, etc (Penz et al., 1996).

Monitorização indirecta

A monitorização indirecta envolve um conjunto de informação quantitativa e qualitativa que, por apenas estar indirectamente relacionada com as acções dos utilizadores sobre o sistema, não se enquadra nos tipos de monitorização descritos anteriormente. Apresentam-se dois casos concretos de monitorização indirecta:

- Indicadores de utilização do sistema – Permitem analisar a utilização do sistema. Por exemplo, o sistema pode analisar a quantidade e dimensão das mensagens trocadas pelos utilizadores para medir e reportar o seu grau de participação (Ackerman, 1995). Um outro exemplo é o fornecimento de uma medida da relevância da interacção dos utilizadores do sistema, a partir da análise do conteúdo das mensagens trocadas. Esta análise de conteúdo pode consistir numa contabilização da frequência de palavras e comparação com uma lista de palavras relevantes;
- Indicadores do comportamento do sistema – Disponibilizam informação sobre o comportamento do sistema e sobre a influência desse comportamento na sua utilização. Por exemplo, Sarin e Greif (1985) utilizam um mecanismo de tele-ponteiro que, para além de mostrar ao utilizador o movimento local do tele-ponteiro (monitorização convergente), mostra igualmente o movimento de um outro objecto gráfico que caracteriza a posição do tele-ponteiro apercebida pelos restantes utilizadores. Num sistema em que a comunicação dos movimentos do tele-ponteiro sofre um atraso arbitrário, este tipo de monitorização possibilita que o utilizador do tele-ponteiro se acomode ao atraso com que os restantes utilizadores do sistema o visualizam.

7.4 Mecanismos configuráveis

Diversos sistemas cooperativos permitem que os utilizadores registem o interesse em serem notificados quando determinados eventos ocorrerem no

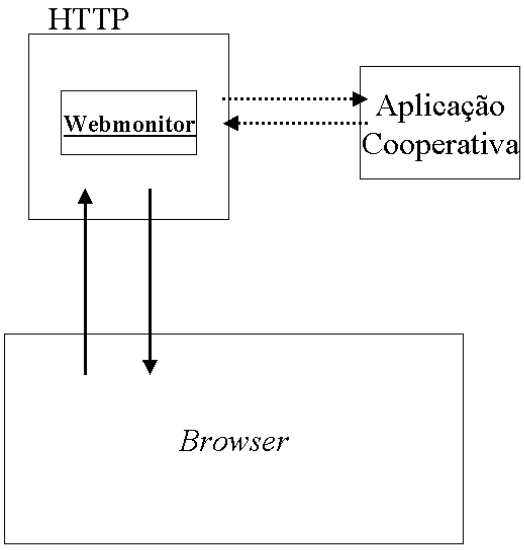


Figura 7.9: Arquitectura do WebMonitor

sistema. O objectivo é possibilitar a configuração individual do grau ou tipo de monitorização do sistema. No entanto, a maior parte destes sistemas apenas fornece mecanismos básicos de distribuição de eventos, deixando para o programador da aplicação a definição de mecanismos apropriados de interface pessoa-máquina que suportem a monitorização.

Há dois aspectos importantes a considerar na notificação de eventos (Berlage e Sohlenkamp, 1999):

- A notificação deve ser selectiva e depender do destinatário, de forma a evitar o excesso de informação. Logo, devem ser implementados mecanismos de filtragem, agregação e pesagem de eventos.
- A apresentação dos eventos deve considerar aspectos como a relevância, visibilidade e receptividade do destinatário. Por vezes é necessário que o sistema force o destinatário a monitorizar o evento.

7.5 Exemplos

Webmonitor

Uma questão paradoxal associada à navegação na WWW é que sendo possível partilhar uma imensidão de informação por uma imensidão de pessoas, o acto de navegar é solitário.

Do ponto de vista do suporte a cooperação, a questão fundamental é que os *browsers* são unidireccionais. Um exemplo deste defeito é o caso em que

um utilizador altera uma determinada página da WWW mas as suas modificações não podem ser observadas simultaneamente pelos restantes utilizadores que nesse momento a estão a observar ou manipular.

O Webmonitor (Oliveira e Ferraz, 1998) é uma plataforma de serviços para colaboração em grupo que transforma a WWW de um armazém primário de informação passiva numa ferramenta de cooperação activa. O seu papel é o de monitorizar as actividades dos utilizadores quando navegam na WWW. Desta forma, as páginas tornam-se salas de convívio onde se pode entrar, sair, encontrar e interactuar com outras pessoas, sejam elas conhecidas ou não.

O Webmonitor oferece uma interface padronizada para monitorização das actividades dos utilizadores sobre uma aplicação cooperativa ou sobre uma página HTML disponibilizada na WWW. O Webmonitor disponibiliza uma janela de mensagens, onde aparecem comunicações de outros utilizadores ou do servidor, uma janela de utilizadores activos que indica quem está no local, outra semelhante mas que indica quem está no documento

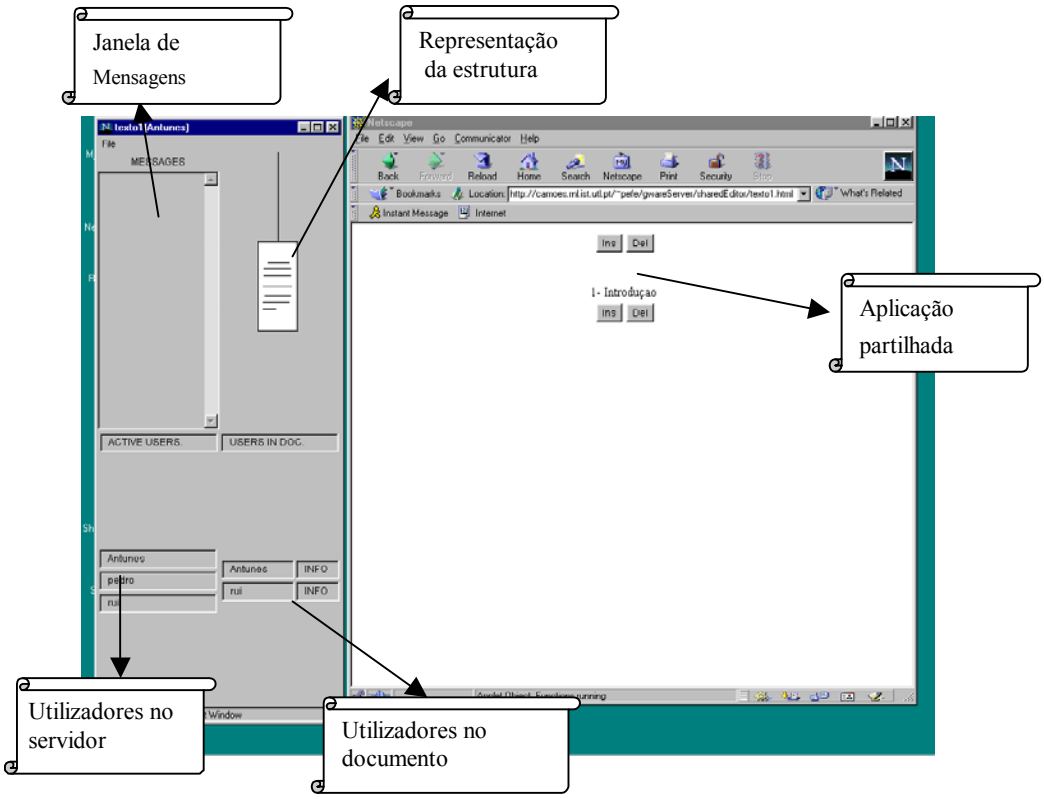


Figura 7.10: Webmonitor

seleccionado e, finalmente, uma janela de estrutura onde se pode visualizar, caso o nível de operacionalidade o permita, a estrutura do documento.

Na Figura 7.9 descreve-se a interacção entre as diversas entidades envolvidas no sistema enquanto na Figura 7.10 se apresenta a interface gráfica do Webmonitor.

Os utilizadores interactuam com o servidor Webmonitor através do seu *browser* habitual, via servidor HTTP. O servidor Webmonitor disponibiliza páginas HTML e partilha diversas aplicações, como por exemplo editores de texto, folhas de cálculo, editores de imagens, etc. Para além disso, o servidor Webmonitor disponibiliza informação sobre os restantes utilizadores que simultaneamente lhe acedem e permite o diálogo entre esses utilizadores.

A comunicação entre o servidor Webmonitor e a aplicação cooperativa é realizada em três níveis de integração possíveis, dependendo das características da aplicação cooperativa:

- Nível 0 – Nível de funcionalidade reduzida, destinado a aplicações que não tenham qualquer tipo de interacção com o servidor Webmonitor. O servidor Webmonitor lança a aplicação e deixa de haver interacção entre ambos. A partilha de páginas HTML estáticas enquadra-se neste nível. A este nível, o Webmonitor apenas pode disponibilizar informação sobre a identidade dos utilizadores que acedem simultaneamente à aplicação ou página HTML;
- Nível 1 – A aplicação serve-se da API do Webmonitor para enviar mensagens para os utilizadores, informando-os acerca de eventos relacionados com o documento visualizado;
- Nível 2 – Além das mensagens de nível 1, a aplicação disponibiliza também informação referente à estrutura do documento. O utilizador tem assim informação acerca de como o documento está a ser alterado no que se refere à sua estrutura.

Como se pode verificar, o sistema é composto por duas entidades separadas, mas que podem interligar-se entre si com diferentes níveis de complexidade.

A opção por estes três níveis serve para uma melhor integração com aplicações genéricas. Por exemplo, foi integrado um calendário, que possui um nível de integração zero, e um editor cooperativo, já de nível dois.

O protocolo utilizado nas interacções entre o Webmonitor e a aplicação é constituído por um conjunto simples e curto de mensagens apresentado na Tabela 7.1.

Aplicação → Webmonitor:

STATUS# - Verifica o estado do Servidor.
REGISTER#DocName#CliID#AppName# - Regista um novo cliente num documento gerido por uma aplicação.
LISTENING#DocName#CliID# - Resincroniza um cliente no documento.
SENDMESSAGE#CliID#DestinCli#Msg# - Envia uma mensagem para um utilizador.
BRDCSTMSGDOC#DocName#CliID#Msg# - Envia uma mensagem para todos os utilizadores que se encontram num mesmo documento.
BRDCSTMESSAGE#CliID#Msg# - Envia uma mensagem para todos os utilizadores.
RELOAD#DocName# - Avisa o Servidor para que os clientes releiam o documento.
RMVACTVUSER#DocName#CliID# - Remove um utilizador do Servidor.
RMVDOCVUSER#DocName#CliID# - Remove um utilizador de um documento.
STRUCT_LIST#DocName#CliID# - Devolve a estrutura do documento (lista).
INSERT#DocName#CliID#Pos# - Insere um novo elemento.
DELETE#DocName#CliID#Pos# - Apaga um elemento.
INFOUSRDOC#DocName#CliID# - Devolve a informação sobre um utilizador.
OPENDOCLIST# - Devolve a lista de documentos abertos.

Webmonitor → Browser:

RCVMESSAGE#Sender#Msg# - Recebe uma mensagem da Aplicação.
ADDACTVUSER#CliID# - Adiciona um utilizador activo.
ADDDOCUSER#CliID# - Adiciona um utilizador ao documento.
RMVACTVUSER#CliID# - Remove um utilizador activo.
RMVDOCUSER#CliID# - Remove um utilizador do documento.
RELOAD# - Ordem para reler o documento.
STRUCT_LIST#StructVector#imageURL# - Informa sobre a estrutura do documento.

Tabela 7.1: Mensagens utilizadas pelo Webmonitor

A comunicação entre o *browser* e o servidor Webmonitor é implementada por uma *applet java* que estabelece uma ligação TCP/IP com o servidor.

FaceTalk

O FaceTalk (Penz et al., 1996) é uma aplicação de conversação (*chat*) típica a que foi adicionado um mecanismo que permite aos utilizadores transmitirem emoções sobre a mensagem que estão a ler ou escrever. Para definir as emoções o utilizador pode controlar quatro parâmetros (Figura 7.11): olhos, sobrolhos, boca e orelhas. Este é portanto um exemplo de monitorização do grupo de trabalho.

IdeaGen

O IdeaGen apresenta um conjunto de objectos gráficos, designados monitores, que dão pistas sobre as actividades que os utilizadores exercem

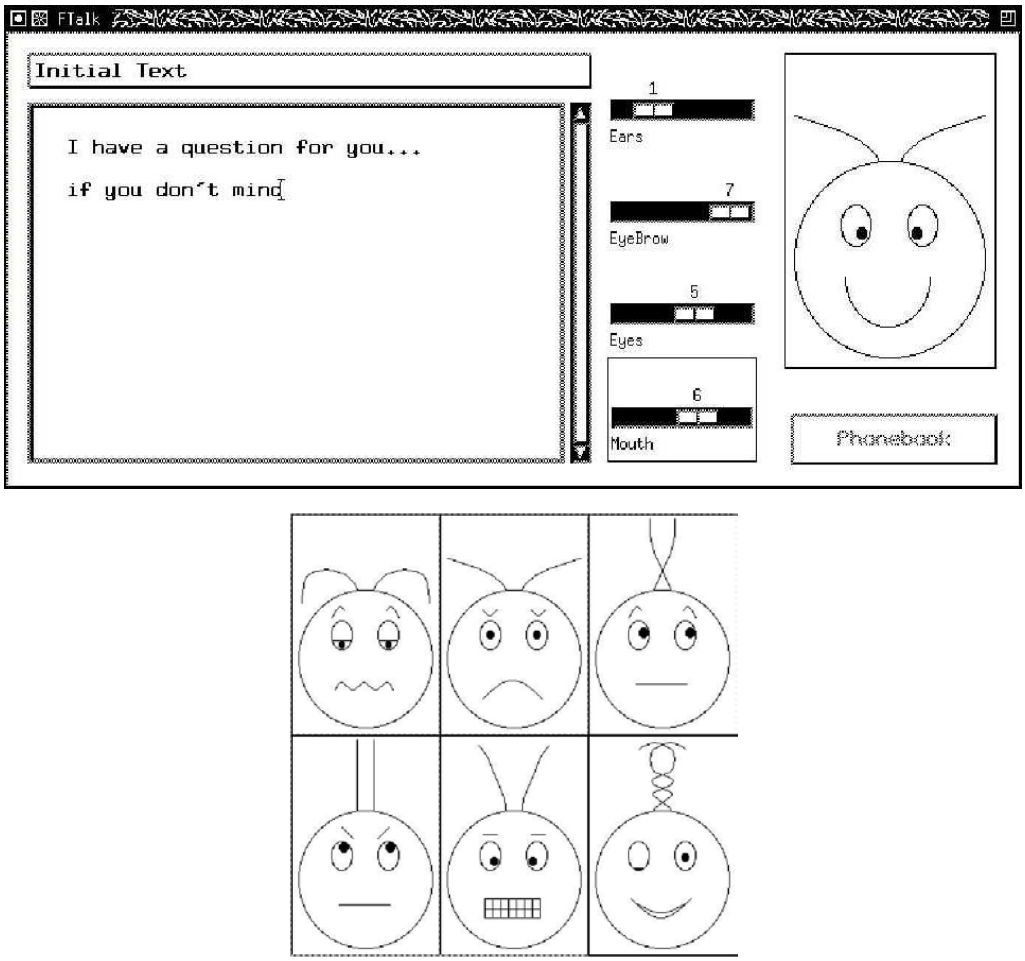


Figura 7.11: FaceTalk: Mecanismo de composição de emoções e variantes possíveis

sobre os objectos presentes no espaço público. Os monitores são criados ao nível da interface com o utilizador, junto dos objectos públicos.

Na ausência de eventos interessantes um monitor não é visível, de forma a não introduzir elementos distractivos na interface.

Os monitores suportados pelo IdeaGen são:

- Monitor de concorrência – Fornece indicações aos utilizadores sobre o estado do protocolo de acesso aos objectos públicos;
- Monitor de propriedade – Permite visualizar as propriedades dos objectos públicos, quando estas não podem ser inferidas directamente a partir dos próprios objectos (porque não estão visíveis);

Figura 7.12a: Monitorização de propriedade: Pionés indica se o objecto pode ser movido

Figura 7.12b: Monitorização de estado: Lupa indica que o objecto foi modificado

Figura 7.12c: Monitorização de elasticidade temporal: Barra indica progresso do sistema

Figura 7.12d: Monitorização de concorrência: Semáforo indica o estado do protocolo

- Monitor de elasticidade temporal – Monitoriza o estado do modelo WYGIWIG utilizado pelo espaço público;
- Monitor de gestão de interface – Monitoriza o estado do modelo WYSIWIMS também utilizado pelo espaço público. Como este modelo permite que os utilizadores mantenham vistas incoerentes sobre os mesmos dados, este monitor serve para avisar das alterações às estruturas de dados que não sejam directamente observáveis no espaço público.

O monitor de propriedade aparece junto a um objecto no espaço público e recorre à metáfora do "pionés" para expressar a propriedade desse objecto (Figura 7.12a). O pionés indica se o objecto pode ou não ser movido no espaço de trabalho.

O monitor de gestão da interface permite que um objecto no espaço público indique se algum dos seus dependentes não visíveis para o utilizador (modelo WYSIWIMS) foi modificado por outros utilizadores (Figura 7.12b). Este monitor aparece quando essas alterações acontecem e desaparece quando o utilizador torna os dependentes visíveis.

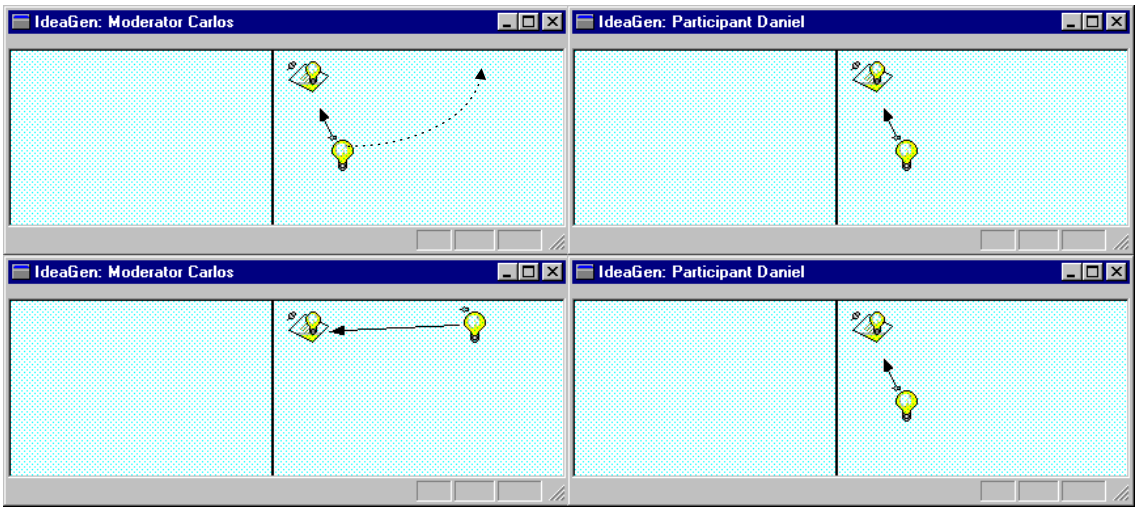


Figura 7.13a: Utilização do monitor de propriedade: Utilizadores sabem que podem mover o objecto

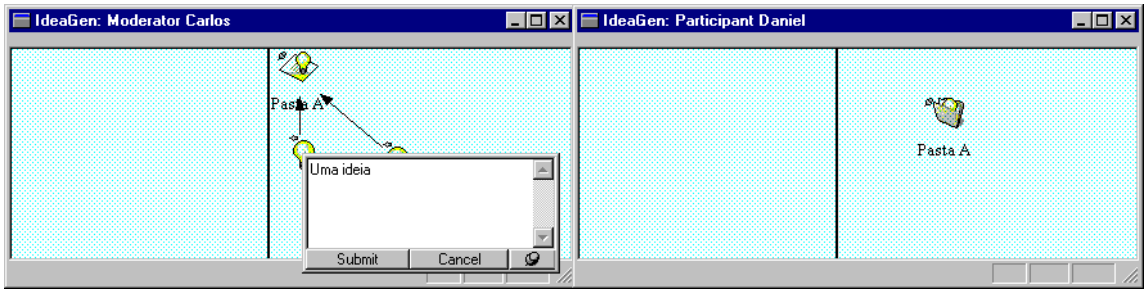


Figura 7.13b: Utilizador da esquerda edita um objecto público

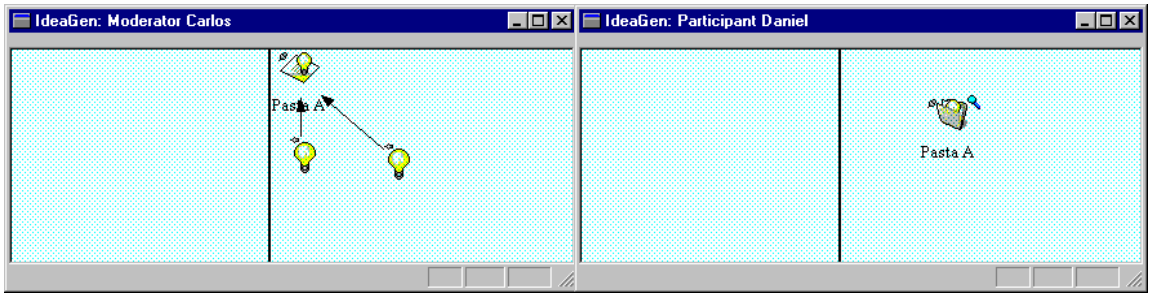


Figura 7.13c: Utilizador da direita é informado da modificação pelo monitor de gestão da interface

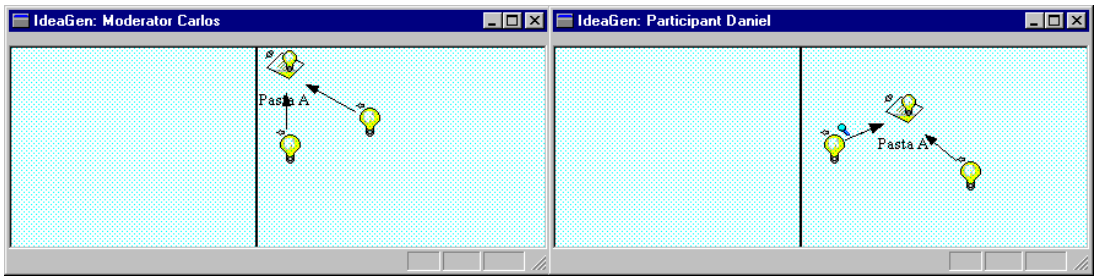


Figura 7.13d: Utilizador abre a pasta para verificar as modificações. O monitor indica que a modificação ainda não está visível

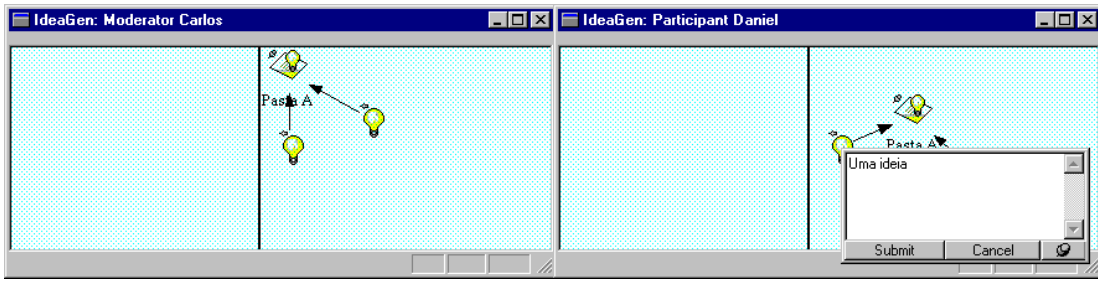


Figura 7.13e: Finalmente o utilizador observa a modificação

O monitor de elasticidade temporal permite tomar consciência da quantidade de membros de um grupo que já visualizaram ou realizaram uma determinada tarefa. Para tal, este monitor recorre a uma barra de progresso (Figura 7.12c).

O monitor de concorrência indica a ocorrência, ou possibilidade de ocorrência, de conflitos no acesso aos objectos públicos. A metáfora utilizada é a de um semáforo (Figura 7.12d).

Na Figura 7.13 ilustra-se a utilização destes monitores pela ferramenta.

8 Acesso

O controlo de acesso opera em dois níveis distintos:

- Objectos – Determina quem pode aceder aos objectos partilhados do sistema e quando. O controlo de acesso não pode ser confundido com o controlo da concorrência dos objectos, dado que se destina a organizar as tarefas dos utilizadores, sobrepondo-se assim ao controlo das tarefas concorrentes sobre os objectos.
- Sessão – Determina quem pode aceder ao sistema cooperativo, gerindo todo o processo de entrada e saída.

O controlo de acesso dos objectos partilhados do sistema realiza-se normalmente a partir de especificações dos atributos desses objectos, por exemplo visualizar, modificar, mover, e dos respectivos privilégios de cada utilizador. É natural que os objectos estejam organizados hierarquicamente e nesse caso a atribuição de privilégios será propagada ao longo da hierarquia de objectos. Por exemplo, no caso do Calliope, um editor de texto multi-utilizador, o acesso pode ser restringido ao nível da palavra, linha, parágrafo ou documento (Greenberg e Roseman, 1996).

O controlo de sessão procura oferecer um processo flexível de ligação dos utilizadores ao sistema, permitindo que estes cheguem tarde mas possam recuperar o estado do sistema, ou possam sair a meio para retomarem a sessão mais tarde. O controlo da sessão envolve as seguintes actividades (Greenberg e Roseman, 1996):

- Criação de sessões;
- Identificação de sessões;
- Remoção de sessões;
- Localização de sessões que se encontrem a decorrer;
- Identificação dos utilizadores ligados a uma sessão;
- Gestão da lista dos utilizadores que se podem ligar a uma sessão;
- Convite à participação numa sessão;
- Controlo de acesso a uma sessão;
- Ligação a uma sessão;
- Ligação a meio da sessão;
- Saída de sessão;
- Manutenção de informação persistente sobre a sessão.

O sistema, para além do suporte às actividades descritas acima, deve implementar políticas de gestão de sessões. Exemplos deste tipo de políticas são as políticas de *log-in*, porta aberta, convite e *rendezvous*. A política de *log-in* permite que um utilizador se ligue em qualquer momento à sessão, sendo requerida a indicação de um nome de utilizador e uma palavra de passe. A política de porta aberta permite que qualquer utilizador se ligue a uma sessão. Ao contrário, na política de convite apenas os utilizadores previamente convidados pelo dono da sessão o podem fazer.

A política de *rendezvous* é semelhante à política de porta aberta mas oferece aos utilizadores uma imagem estruturada das sessões que se encontram disponíveis e dos utilizadores que a elas se encontram ligados (por exemplo, os MUDs oferecem esta funcionalidade). Outras designações têm sido atribuídas a esta política, como *browsing* social, ou interacção casual (kristoffersen, 1998). O conceito por detrás destas designações é que o sistemas cooperativo deve facilitar os encontros casuais entre os utilizadores.

9 Blocos construtivos

A discussão realizada neste documento resultou na identificação e sistematização de diversos blocos construtivos necessários ao desenho e realização de sistemas cooperativos. Os blocos construtivos considerados foram:

- Comunicação
- Arquitectura
- Concorrência
- Coordenação
- Espaço público
- Monitorização
- Acesso

Irei agora propor um enquadramento lógico destes blocos construtivos. Como se pode observar na Figura 9.1, o esquema proposto identifica um conjunto significativo de objectos organizados em torno de três malhas lógicas.

A primeira malha a considerar é destinada a definir a arquitectura do sistema e os modelos de comunicação e concorrência, integrando os seguintes objectos:

IU – Objectos públicos de interface com os utilizadores.

S – Objectos contendo a semântica da aplicação.

Gcc – Gestores da concorrência, que implementam os mecanismos e protocolos necessários à manutenção da coerência dos dados do sistema (ou seja, dos objectos S). Os protocolos podem ser optimistas, pessimistas ou uma combinação dos dois (fragmentados).

Gcm – Gestores de comunicação, responsáveis pela comunicação entre os objectos definidos pela arquitectura do sistema. Permitem comunicação ponto-a-ponto, multiponto e difusão. A comunicação pode ser síncrona ou assínocrona. No caso de comunicação assíncrona, os gestores são também responsáveis por guardar as mensagens. Os gestores de comunicação podem lidar com canais independentes ou interdependentes.

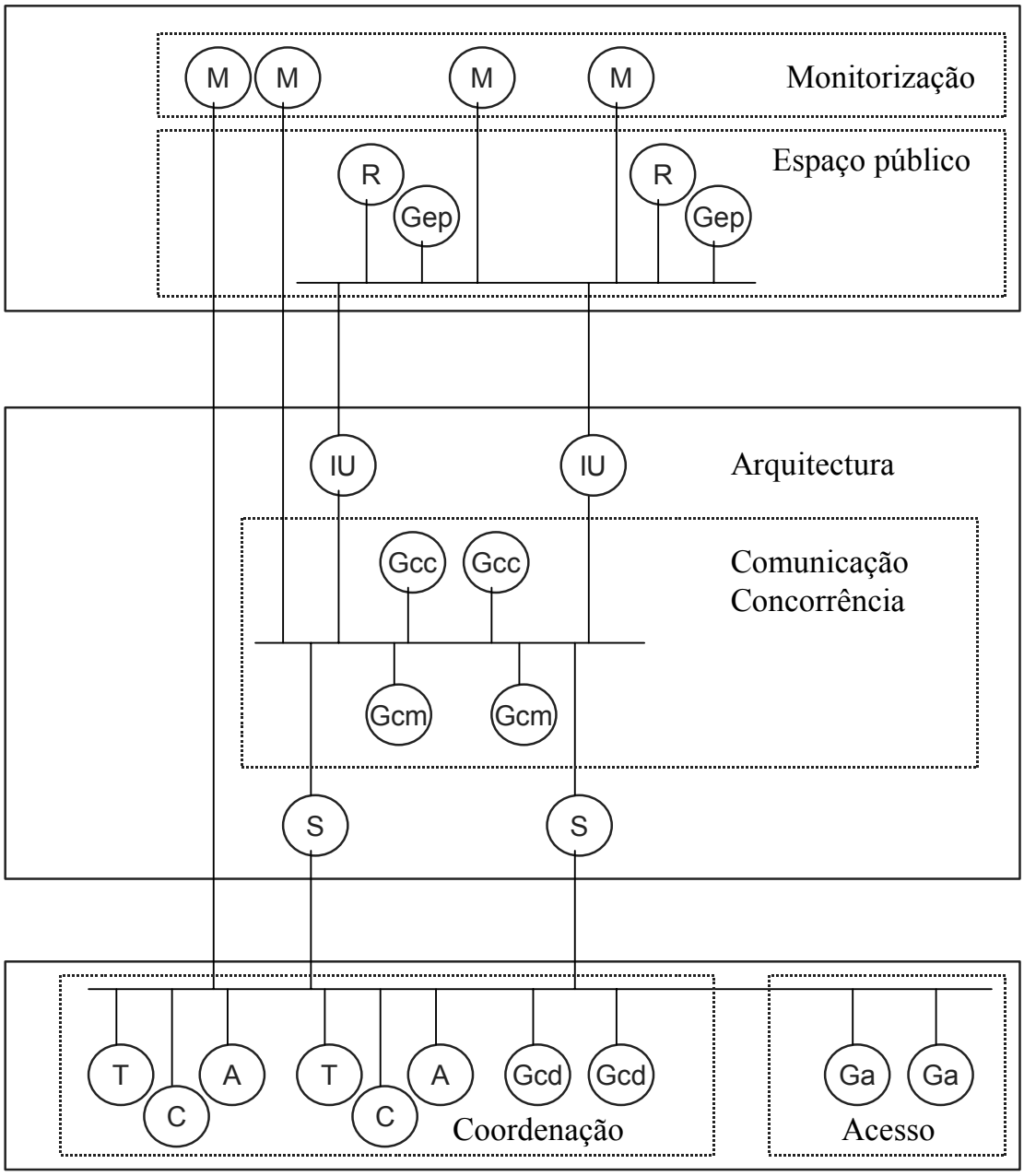


Figura 9.1: Esquema global dos blocos construtivos de um sistema de suporte a cooperação

A partir deste conjunto de objectos é possível criar variantes que correspondam às diferentes arquitecturas que foram por nós consideradas. Por exemplo, o conjunto de objectos {S, Gcc, Gcm, IU_1...IU_n} corresponde a uma arquitectura centralizada mono-utilizador ou multi-utilizador.

O conjunto {{S_1, Gcm_1,Gcc_1}...{S_n, Gcm_n,Gcc_n}} corresponde a uma arquitectura distribuída. O conjunto {Gcc, {S_1, Gcm_1}...{S_n, Gcm_n}} corresponde a uma arquitectura híbrida.

A segunda malha lógica a considerar destina-se a materializar a noção de espaço público e interliga os seguintes objectos:

R – Correspondem às réplicas dos objectos públicos IU que são criadas no nó de cada utilizador.

M – Objectos que implementam os mecanismos de monitorização das interacções, sejam do espaço de trabalho, monitorização da equipa de trabalho ou monitorização indirecta.

Gep – Definem o modelo de espaço público, a partir do controlo que exercem sobre os objectos R.

IU – Os objectos IU encapsulam e definem a funcionalidade global dos objectos R, M e Gep existentes no sistema.

O conjunto de objectos {Gep, R_1...R_n} permite definir uma vista sobre o espaço público virtual. Outra funcionalidade possível para o conjunto {Gep, R_1...R_n} é definir quais os atributos de R_1...R_n que são apresentados em modo estrito.

Quanto aos objectos M, note-se que a diversidade de mecanismos de monitorização da interacção dos utilizadores do sistema cooperativo exige que estes possam aceder às três malhas lógicas.

Finalmente, a terceira malha lógica engloba a coordenação e o controlo de acesso dos utilizadores do sistema, abrangendo os seguintes objectos:

T – Suportam tarefas dos utilizadores.

C – Suportam conversações entre os utilizadores.

A – Implementam artefactos que servem de mediadores entre os participantes (tipicamente encapsulam e definem a funcionalidade global de diversos objectos do tipo S e IU).

Gcd – Definem o modelo de coordenação dos utilizadores do sistema, a partir do controlo que exercem sobre os objectos T, C e A.

Ga – Fazem a gestão de acesso, quer da sessão quer dos objectos S. Dado que os objectos do tipo A encapsulam objectos do tipo S, então os objectos Ga também podem gerir o acesso a objectos do tipo A.

S – Os objectos S encapsulam a semântica da aplicação cooperativa.

Observe-se que a coordenação indirecta pode ser realizada com base no conjunto de objectos {Gcd, A_1...A_n}, sendo Gcd responsável por definir as regras de acesso aos artefactos A.

A coordenação sequencial formal pode ser concretizada com o conjunto de objectos {Gcd, T_1...T_n}, operando Gcd como escalonador de tarefas. A coordenação sequencial semi-formal pode recorrer ao conjunto {{Gcd_1, T_1}...{Gcd_n, T_n}}, sendo que, neste caso, os objectos Gcd servem de filtros às mensagens trocadas entre objectos T.

A coordenação recíproca pode recorrer ao conjunto de objectos {Gcd, C_1...C_n}. Por exemplo, nos casos em que pretenda recorrer aos modelos IBIS ou actos de fala, esses modelos podem ser especificados por Gcd.

A gestão de acesso a uma hierarquia de objectos pode ser concretizada com o conjunto de objectos {A, Ga, {S_1, Ga_1}... {S_n, Ga_n}}.

Note-se ainda que os objectos S têm uma função dual, contendo quer os dados quer a semântica da aplicação.

Agradecimentos

Este trabalho foi realizado no âmbito do projecto MOOSCO (MOOS With Separation of Concerns), financiado pela Fundação para a Ciência e Tecnologia (POSI/CHS/33127/99).

Bibliografia

M. Ackerman and B. Starr. Social activity indicators: interface components for CSCW systems. Proceedings of the 8th ACM symposium on User interface and software technology, 1995.

A. Agostini, G. De Michelis, S. Patriarca, and R. Tinini. A prototype of an integrated coordination support system. Computer Supported Cooperative Work, 2, 1994.

A. Agostini, G. De Michelis, and M. Grasso. Rethinking CSCW systems: the architecture of Milano. In Proceddings of the Fifth European Conference on Computer Supported Cooperative Work. Kluwer Academic Publishers, Lancaster, UK, September 1997.

S. Ahuja, J. Ensor, and S. Lucco. A comparison of applications sharing mechanisms in real-time desktop conferencing systems. In Proceedings of the Conference on Office Information Systems, Boston, 1990.

C. Alves and D. Silva. Suporte e Controlo da Interacção em Sistemas de Trabalho Cooperativo. Trabalho Final de Curso. IST. Setembro 1997.

P. Antunes, N. Guimaraes, and R. Nunes. Extending the user interface to the multiuser environment. In ACM SIGOIS Bulletin. April 1992.

P. Antunes and N. Guimarães. A Distributed Model and Architecture for Interactive Cooperation. In Proceedings of the 4th Workshop on Future Trends of Distributed Computing Systems, FTDCS '93. Lisboa, Portugal: IEEE CS Press, 1993.

P. Antunes and N. Guimarães. User-Interface Support to Group Interaction. In Second International Workshop on Groupware, CRIWG '96. Puerto Varas, Chile, September, 1996.

P. Antunes. A System for Supporting and Managing Same-Time/Different-Place Group Interactions. In Proceedings of the Working Conference on Advanced Visual Interfaces, AVI '98. L'Aquila, Italy: ACM Press, 1998.

W. Appelt. What groupware functionality do users really use? Proceedings of the 9th Euromicro Workshop on PDP 2001. Mantua, February, 2001. IEEE Computer Society, Los Alamitos.

L. Bannon. CSCW: An initial exploration. Scandinavian Journal of Information Systems, 5. August 1993.

N. Barghouti and G. Kaiser. Concurrency control in advanced database systems. ACM Computing Surveys, 23(3):269-317, September 1991.

N. Barrocas. Ferramentas de Apoio ao Planeamento e Condução de Reuniões Electrónicas. Trabalho Final de Curso. IST. Outubro 1999.

M. Beaudouin-Lafon and A. Karsenty. Transparency and awareness in a real-time groupware system. In Proceedings of the ACM Symposium on User Interface Software and Technology, Monterey, California, November 1992.

S. Benford. A spatial model of interaction in large virtual environments. In Proceedings of the Third European Conference on Computer-Supported Cooperative Work - ECSCW '93, Milan, September 1993.

R. Bentley, T. Rodden, P. Sawyer, and I. Sommerville. An architecture for tailoring cooperative multi-user displays. In Proceedings of ACM CSCW '92 Conference on Computer-Supported Cooperative Work, Toronto, Canada, November 1992.

R. Bentley, T. Rodden, P. Sawyer, and I. Sommerville. Architectural support for cooperative multiuser interfaces. IEEE Computer, May 1994.

T. Berlage and M. Sohlenkamp. Visualizing common artefacts to support awareness in computer-mediated cooperation. Computer Supported Cooperative Work, 8. 1999.

M. Bragen. Go with the flow. PC Magazine, June 14 1994.

L. Brothers, V. Sembugamoorthy, and M. Muller. Icicle: Groupware for code inspection. In Proceedings of the Conference on Computer Supported Cooperative Work (CSCW '90), Los Angeles, California, 1990.

R. Butler. Designing Organizations, chapter Requisite Decision-Making Capacity. Routledge, 1991.

J. Conklin. Hypertext: An introduction and survey. In I. Greif, editor, Computer-Supported Cooperative Work: a Book of Readings, chapter 16. Morgan Kaufmann Publishers Inc, 1988.

S. Cook, G. Birch, G. Birch, A. Murphy, and J. Woolsey. Modelling groupware in the electronic office. In S. Greenberg, editor, Computer Supported Collaborative Work. Academic Press, Inc., 1991.

F. Cosquer, P. Antunes, and P. Veríssimo. Enhancing Dependability of Cooperative Applications in Partitionable Environments. In Proceedings of the 2nd European Dependable Computing Conference, EDCC-2. Taormina, Italy: Lecture Notes in Computer Science, Springer-Verlag, 1996.

T. Crowley, E. Baker, H. Forsdick, P. Milazzo, and R. Tomlinson. Mmconf: an infrastructure for building shared applications. In Proceedings of the Conference on Computer Supported Cooperative Work (CSCW '90), Los Angeles, California, 1990.

A. Dennis, J. Valacich, C. Speier, and M. Morris. Beyond media richness: An emprirical test of media synchronicity theory. Proceedings of the 31st Hawaii International Conference on System Sciences. Hawaii, 1998.

P. Dewan. Flexible user interface coupling in collaborative systems. In ACM SIGCHI Conference on Human Factors in Computing Systems, pages 41-48, New Orleans. ACM Press, 1991.

P. Dourish and V. Bellotti. Awareness and coordination in shared workspaces. In Proceedings of ACM CSCW '92 Conference on Computer-Supported Cooperative Work, pages 107-114, Toronto, Canada, November 1992.

C. Ellis, S. Gibbs, and G. Rein. Groupware: Some issues and experiences. Communications of the ACM, 34(1):38-58, 1991.

C. Ellis and J. Wainer. Goal-based models of collaboration. Collaborative Computing, 1:61-86, 1994.

S. Elrod, R. Bruce, R. Gold, D. Goldberg, F. Halasz, W. Janssen, D. Lee, K. McCall, E. Pedersen, K. Pier, J. Tang, B. Welch. Liveboard: a large interactive display supporting group meetings, presentations, and remote collaboration. Conference proceedings on Human factors in computing systems. June 1992.

M. Elwart-Keys, D. Halonen, M. horton, R. Kass, and P. Scott. User interface requirements for face to face groupware. In CHI '90: Conference on Human Factors in Computing Systems, April 1990.

D. Engelbart. Authorship provisions in augment. In I. Greif, editor, Computer-Supported Cooperative Work: a Book of Readings, chapter 5. Morgan Kaufmann Publishers Inc, 1988a.

D. Engelbart and W. English. A research center for augmenting human intellect. In Computer-Supported Cooperative Work: a Book of Readings, chapter 4. Morgan Kaufmann Publishers Inc, 1988b.

T. Erikson and W. Kellogg. Social translucence: An approach to designing systems that support social processes. ACM transactions on Computer-Human Interaction, 7(1). March 2000.

F. Flores, M. Graves, B. Hartfield, and T. Winograd. Computer systems and the design of organizational interaction. ACM Transactions on Office Information Systems, 6(2):153-172, 1988.

S. Greenberg. Sharing views and interactions with single-user applications. In Proceedings of the Conference on Office Information Systems, pages 227-237, Boston, 1990.

S. Greenberg and M. Roseman. Groupware toolkits for synchronous work. Trends in CSCW. M. Beaudouin-Lafon (ed.). John Wiley & Sons. 1996.

S. Greenberg. Peepholes: Low Cost Awareness of One's Community. In CHI '96: Conference on Human Factors in Computing Systems. Vancouver, Canada, 1996.

S. Greenberg and C. Gutwin. From Technically Possible to Socially Natural Groupware. In Proceedings of the 9th NEC Research Symposium: The Human-centric Multimedia Community. Nara, Japan. August 1998.

S. Greenberg and H. Kuzuoka. Bootstrapping Intimate Collaborators. In OzCHI99 Workshop: Issues of Use in CSCW Technology Design (held as part of the OZCHI'99 Australian Conference on Computer Human Interaction). 1999.

J. Grudin. Computer-supported cooperative work: History and focus. IEEE Computer. 1994.

C. Gutwin, S. Greenberg, and M. Roseman. Workspace awareness support with radar views. Proceedings of the CHI '96 conference companion on Human factors in computing systems. April 1996.

C. Gutwin and S. Greenberg. Design for individuals, design for groups: Tradeoffs between power and workspace awareness. Proceedings of the 1998 ACM conference on Computer supported cooperative work. November 1998.

C. Gutwin and S. Greenberg. The effects of workspace awareness support on the usability of real-time distributed groupware. ACM Transactions on Computer-Human Interaction, 6(3). September 1999.

A. Haake and J. Haake. Take CoVer: Exploiting version support in cooperative systems. In Human Factors in Computing Systems INTERCHI '93 Conference Proceedings, Amsterdam. Addison-Wesley, April 1993.

C. Hwang and M. Lin. Group Decision Making under Multiple Criteria: Methods and Applications. Springer-Verlag, 1987.

H. Ishii, M. Kobayashi, and K. Arita. Iterative design of seamless collaboration media. Communications of the ACM, 37(8), August 1994.

R. Johansen, D. Sibbet, S. Benson, A. Martin, R. Mittman, and P. Saffo, *Leading business teams*. Addison-Wesley, 1991.

N. Kamel. An integrated approach to shared synchronous groupware workspaces. In Proceedings of the 4th Workshop on Future Trends of Distributed Computing Systems, Lisboa, Portugal, September 1993.

S. Kaplan, A. Carrol, and K. MacGregor. Supporting collaborative processes with ConversationBuilder. In Conference on Organizational Computing Systems, pages 69-79, Atlanta, Georgia, November 1991.

S. Kaplan, W. Tolone, D. Bogia, and C. Bignoli. Flexible, active support for collaborative work with ConversationBuilder. In Proceedings of ACM CSCW '92 Conference on Computer-Supported Cooperative Work, pages 378-385, Toronto, Canada, November 1992.

R. Karbe and N. Ramsperger. Influence of exception handling on the support of cooperative office work. In Proceedings of IFIP WG8.4 Conference on Multi-User Interfaces and Applications, Crete. North-Holland, 1990.

F. Kon, F. Costa, G. Blair, and R. Campbell. The case for reflective middleware. Communications of ACM, June 2002.

S. Kristoffersen. Developing collaborative multimedia: The MEDIATE toolkit. Phd Thesis. Lancaster University. July 1998.

J. Lauwers and K. Lantz. Collaboration awareness in support of collaboration transparency: Requirements for the next generation of shared

window systems. In CHI '90: Conference on Human Factors in Computing Systems, Seattle, Washington. ACM Press, 1990.

J. Lauwers, T. Joseph, K. Lantz, and A. Romanow. Replicated architectures for shared window systems: a critique. In Proceedings of the Conference on Office Information Systems, Boston, 1990.

I. Lu and M. Mantei. Idea management in a shared drawing tool. In Proceedings of the Second European Conference on Computer Supported Cooperative Work - ECSCW '91, pages 97 112, Amsterdam, 1991.

T. Malone, K. Grant, K-Y. Lai, R. Rao, and D. Rosenblitt. Semi-structured messages are surprisingly useful for computer-supported coordination. ACM Transactions on Office Information Systems, 5(2):115-131, 1987.

T. Malone, K. Grant, K-Y. Lai, R. Rao, and D. Rosenblitt. The information lens: An intelligent system for information sharing and coordination. In Ronald M. Baecker, editor, Readings in Groupware and Computer-Supported Cooperative Work, pages 461-473. Morgan Kaufmann Publishers Inc, 1993.

T. Malone and K. Crowston. The Interdisciplinary Study of Coordination. ACM Computing Surveys, vol. 26, no. 1. March 1994.

G. De Michelis. Situating conversations within the language/action perspective: The milan conversation model. In ACM 1994 Conference on Computer Supported Cooperative Work CSCW '94, Chapel Hill, North Carolina, October 1994.

D. Miller, J. Smith, and M. Muller. TelePICTIVE: Computer supported collaborative GUI, design for designers with diverse expertise. In Proceedings of the ACM Symposium on User Interface Software and Technology, Monterey, November 1992.

H. Mintzberg. Structure in Fives. Prentice-Hall, 1993.

J. Nunamaker, A. Dennis, J. Valacich, D. Vogel, and J. George. Electronic meeting systems to support group work. Communications of the ACM, 34(7), July 1991.

S. Ochoa, L. Guerrero, D. Fuller, and O. Herrera. Designing the communications infrastructure of groupware systems. Groupware: Design, Implementation and Use, 8th International Workshop, CRIWG 2002

Proceedings. Springer-Verlag, Lecture Notes in Computer Science, 2440. September 2002.

A. Oliveira and P. Ferraz. Monitorização de Grupo Utilizando Tecnologias WWW. Trabalho Final de Curso. IST. Outubro 1998.

A. Osborn. Applied Imagination. New York: Scribner, 1963.

F. Pacull, A. Sandoz, and A. Schiper. Duplex: A distributed collaborative editing environment in large scale. In ACM 1994 Conference on Computer Supported Cooperative Work CSCW '94, Chapel Hill, North Carolina, October 1994.

B. Patton, K. Giffin, and E. Patton. Decision-Making Group Interaction, chapter Effective Group Decision Making. Harper Collins Publishers, 1989.

F. Penz, P. Antunes, and M. Fonseca. Feedback in Computer Supported Cooperation Systems: Example of the User Interface Design for a Talk-Like Tool. In The Design of Computer Supported Cooperative Work and Groupware Systems, D. Shapiro, M. Tauber, and R. Traunmueller, Eds. Human Factors in Information Technology, North Holland, 1996.

G. Rein and C. Ellis. rIBIS: A real-time group hypertext system. Int. J. Man-Machine Studies, 34(3):349-368, 1991.

M. Robinson, S. Pekkola, and D. Snowdon. Cat's cradle: Working with other people in overlapping real and virtual worlds through tangled strands of visual and other media. Fourth International Workshop on Groupware, CRIWG '98. September 1998.

T. Rodden and G. Blair. CSCW and distributed systems: the problem of control. In Proceedings of the Second European Conference on Computer Supported Cooperative Work - ECSCW '91. Amsterdam, 1991.

M. Roseman and S. Greenberg. GroupKit a groupware toolkit for building real-time conferencing applications. In Proceedings of ACM CSCW '92 Conference on Computer-Supported Cooperative Work, pages 43-50, Toronto, Canada, November 1992.

M. Roseman and S. Greenberg. Building Real-Time Groupware with GroupKit, A Groupware Toolkit. In ACM Transaction on Computer Human Interaction, 3(1), March 1996.

A. Santos and A. Marcos. An algorithm and architecture to support cooperative multimedia editing. In Proceedings of the 4th Workshop on Future Trends of Distributed Computing Systems, Lisboa, Portugal, September 1993.

S. Sarin and I. Greif. Computer-based real-time conferencing systems. IEEE Computer, 18(10), October 1985.

F. Shipman and C. Marshall. Formality considered harmful: Experiences, emerging themes, and directions on the use of formal representations in interactive systems. Computer Supported Cooperative Work, 8. 1999.

C. Simone, G. Mark, and D. Giubbilei. Interoperability as a means of articulation work. Proceedings of the International Joint Conference on Work Activities Coordination and Collaboration, WACC ’99. February 1999.

M. Stefik, G. Foster, D. Bobrow, K. Kahn, S. Lanning, and L. Suchman. Beyond the chalkboard: Computer support for collaboration and problem solving in meetings. Communications of the ACM, 30(1), 1987.

N. Streitz, J. Geissler, J. Haake, and J. Hol. DOLPHIN: Integrated meeting support across local and remote desktop environments and liveboards. In ACM 1994 Conference on Computer Supported Cooperative Work CSCW '94, Chapel Hill, North Carolina, October 1994.

S. Teixeira, P. Vicente, A. Pinto, H. Miranda, L. Rodrigues, J. Martins and A. Rito. Configuring the Communication Middleware to Support Multi-user Object-Oriented Environments. IEEE Proceedings of the International Symposium on Distributed Objects and Applications (DOA). October 2002.

A. Tripathi. Challenges designing next-gemeration middleware systems. Communications of ACM, June 2002.

S. Whittaker and H. Schwarz. Meetings of the board: The impact of scheduling medium on long term group coordination in software development. Computer Supported Cooperative Work, 8. 1999.

T. Winograd and F. Flores. Understanding Computers and Cognition. Addison-Wesley, 1986.

T. Winograd. A Language/Action Perspective on the Design of Cooperative Work. Human-Computer Interaction, 3(1). 1988.