# Cenários de Utilização de Totem

**Arndt von Staa**
**arndt@inf.puc-rio.br**
**Pontifícia Universidade Católica**
**22453 – 900 Rio de Janeiro,**
**Brasil**

*PUC-RioInfMCC 40/95*

**Dezembro 1995**

## Resumo

Neste artigo apresentamos alguns cenários de uso do meta-ambiente de engenharia de software Totem. O artigo tem por meta facilitar a compreensão do objetivo e da utilidade de Totem.

Com o apoio de Totem administradores de ambientes criam e mantém diversos processos e ferramentas de apoio ao desenvolvimento, à manutenção e à gerência do desenvolvimento. Estes elementos estabelecem ambientes de engenharia de software específicos. Esses ambientes específicos apoiam as diversas atividades realizadas ao desenvolver determinada classe de software. De posse do ambiente específico, o gerente de desenvolvimento instancia o comportamento e habilitação das estações de trabalho a serem utilizadas em determinado projeto. Cada estação de trabalho apoia algumas das atividades do desenvolvimento, manutenção e gerência de desenvolvimento de um determinado software. O conjunto de todas as estações de trabalho apoia todas as atividades do desenvolvimento, manutenção e gerência de desenvolvimento deste software.

Neste artigo mostramos como são compostos os ambientes gerados com o auxílio de Totem. Mostramos ainda como ambientes podem ser utilizados como ferramentas CASE simples, ou como ferramentas para o projeto e composição de programas, ou como ferramentas para a geração de programas a partir de especificações. Concluímos com um esboço de arquitetura evidenciando a viabilidade técnica de Totem.

*Palavras chave*: ambientes de engenharia de software, arquitetura de ambientes, requisitos de ambientes.

## Abstract

In this document we present several usage scenarios of Totem. We expect that these scenarios make it easier to understand Totem's goals. Totem is a meta-environment which allows the instantiation of specific development environments. Each of these specific environments supports some of the development process tasks. This article illustrates the components of an environment generated with Totem. It also shows how to use Totem as a simple CASE tool, a complex program design and composition tool and as a program generator. We conclude with a draft of Totem's architecture.

*Keywords*: software engineering environments, environment architecture, requirements for software engineering environments.

# 1. Introdução

Neste documento apresentaremos um esboço do meta-ambiente de engenharia de software assistido por computador Totem, bem como alguns cenários de sua utilização. Estes cenários destinam-se a facilitar a compreensão dos objetivos e da funcionalidade de Totem. Ao final apresentamos um esboço de arquitetura com a finalidade de evidenciar a viabilidade técnica de Totem.

Neste documento não estabelecemos comparações entre as propostas apresentadas na literatura e a proposta de Totem. Este documento objetiva tão somente ilustrar algumas situações de uso consideradas características, complementando outros documentos a respeito de Totem. Em outros documentos serão realizadas comparações da proposta Totem com outras propostas encontradas na literatura.

## 1.1 Definições

Um *ambiente de engenharia de software* é formado por processos de desenvolvimento, métodos de trabalho, heurísticas, técnicas, linguagens de representação (linguagens de documentação e de programação), normas, ferramentas, bibliotecas, estações de trabalho, redes, e bases de dados (repositórios) contendo informações sobre os sistemas desenvolvidos e mantidos. Um ambiente de engenharia de software provê suporte a todas as atividades realizadas ao desenvolver, manter e gerenciar o desenvolvimento de software. Um ambiente é dito *assistido por computador* quando uma parte significativa das atividades se realizam com apoio de computadores interagindo através de algum repositório de dados.

Um *meta-ambiente de engenharia de software* possui ferramentas com as quais se podem criar e manter *ambientes específicos*. Os meta-ambientes são utilizados para definir processos de trabalho, linguagens de representação (formato, sintaxe e semântica de documentos), métodos, técnicas e variadas ferramentas apoiando estes processos, métodos e técnicas. Dependendo da natureza do sistema objetivo a desenvolver e da equipe que realizará o desenvolvimento ou manutenção, configura-se um *ambiente específico* a partir das definições disponíveis no meta-ambiente. Este ambiente específico pode ser particionado em várias estações de trabalho. Cada uma destas estações de trabalho estará habilitada a desempenhar algumas das tarefas do desenvolvimento. Estas estações de trabalho contém as *instâncias de ambiente* geradas.

## 1.2 Objetivos

O *meta ambiente de engenharia de software assistido por computador* Totem tem por objetivos:

- criar e manter ambientes de engenharia de software, capacitando grupos de desenvolvimento e manutenção de software a desenvolver software de alta qualidade com elevada produtividade.
- sintonizar ambientes de modo a aprimorar o desempenho dos grupos de desenvolvimento e manutenção de software.
- criar ambientes para a engenharia reversa e reengenharia de sistemas de software desenvolvidos sem o uso de instâncias de ambiente Totem.
- estabelecer mecanismos para o reúso efetivo[1] em níveis elevados de abstração.

[1] Entendemos por *reúso efetivo* o reúso de componentes, *designs*, arquiteturas ou especificações em que o sistema reutilizador faz uso sem cópias nem modificação de elementos disponíveis em outros sistemas.

- criar e manter ambientes para conduzir pesquisa experimental em engenharia de software. São exemplos de tais pesquisas: a criação e avaliação de linguagens de representação; a criação e avaliação de processos de desenvolvimento e manutenção; a criação e avaliação de transformadores de representação; a criação e avaliação de métodos e técnicas formais de desenvolvimento e manutenção.

O *projeto Totem* tem os seguintes objetivos:

- desenvolver um meta-ambiente de engenharia de software assistido por computador, satisfazendo os objetivos do meta-ambiente definidos acima, e possuindo qualidade pré-industrial[2].
- produzir uma base de dados da história do desenvolvimento de Totem. Esta base dados conterá medições e servirá de instrumento para avaliar o impacto da introdução de ferramentas geradas com o apoio de Totem.
- estabelecer um ambiente de desenvolvimento que satisfaça os requisitos necessários para obter o certificado ISO-9000-3 bem como o nível 3 de maturidade tal como definido no *Capability Maturity Model* do Instituto de Engenharia de Software associado à Universidade de Carnegie-Mellon. Embora o objetivo do projeto não seja obter estes certificados, queremos operar como se tivéssemos eles. Como conseqüência deste objetivo, ambientes gerados com o apoio de Totem possuirão características que permitam obter estes certificados.
- servir de instrumento para a pesquisa em processos de desenvolvimento, métodos de engenharia de software, e em gerência e controle de qualidade. Ou seja, o próprio processo de desenvolvimento de Totem é objeto de pesquisa.
- servir de mecanismo para validar hipóteses relativas ao desenvolvimento sistemático de software de boa qualidade. Ou seja, o processo de desenvolvimento de Totem deverá servir para validar e redefinir os objetivos e requisitos do próprio meta-ambiente Totem.
- servir de laboratório para o desenvolvimento e o ensino de assuntos relacionados com engenharia de software, em particular no tocante ao desenvolvimento de sistemas utilizando técnicas orientadas a objetos e de projeto de componentes.

## 1.3 Resenha do artigo

Na segunda seção descrevemos como Totem é utilizado para gerar ambientes de desenvolvimento. Esta seção introduz diversos conceitos necessários para o entendimento da abrangência de Totem. Na terceira seção descrevemos cenários de uso, desde um simples CASE até um gerador de programas a partir de especificações. Na quarta seção apresentamos um esboço da arquitetura de Totem. Esta seção se destina a ilustrar como Totem poderia ser implementado.

# 2. O Uso de Totem como gerador de ambientes

Nesta seção examinaremos Totem do ponto de vista de uma ferramenta geradora de ambientes de engenharia de software. Estes ambientes destinam-se a apoiar o desenvolvimento de classes de software específicas e são instanciados para operarem em plataformas específicas.

---

2 Entendemos por software de *qualidade pré-industrial* um sistema formado por um conjunto de programas e bases de dados, possuindo elevada confiabilidade e qualidade de engenharia, mas que não necessariamente possui bom empacotamento do ponto de vista de produto. Para alcançar esta qualidade de produto será necessário um investimento complementar.

## 2.1 Características fundamentais de instâncias de ambiente Totem

Totem é um meta-ambiente de engenharia de software assistido por computador. Utilizando o meta-ambiente, definem-se processos, linguagens de representação, regras de transformação entre linguagens e regras de avaliação da qualidade. Esses elementos podem ser configurados formando um ambiente específico. Cada ambiente específico destina-se a apoiar o desenvolvimento de uma dada classe de sistemas. O ambiente específico é então particionado em diversas instâncias de ambiente.

Um ambiente gerado com Totem consta tipicamente de uma coletânea de estações de trabalho, cada uma contendo uma instância de ambiente configurada para apoiar uma ou mais tarefas do desenvolvimento, manutenção ou gerência de desenvolvimento. A configuração de uma instância gerada com Totem é especializada tendo em vista as necessidades dos desenvolvedores, das características dos processos e ferramentas de desenvolvimento utilizados e das características do software a desenvolver ou manter.

O conjunto de instâncias de ambiente apoia o conjunto de todas as tarefas do desenvolvimento. Em particular apoia tarefas: de especificação, arquitetura, projeto modular, projeto detalhado, implementação, geração de documentos, teste, medição e controle de qualidade, empacotamento, controle de evolução, gerência do desenvolvimento, e manutenção.

A figura 1 identifica alguns dos papéis de desenvolvedores (caixas brancas) e de ferramentas (caixas sombreadas) comuns em um ambiente de engenharia de software moderno. Todas essas pessoas e ferramentas interagem uma com as outras. A interação se dá através da troca de informação mantida em um repositório. Em ambientes manuais disciplinados, o repositório é formado por arquivos contendo documentos de papel. Em ambientes indisciplinados a informação é de conhecimento de algumas pessoas e é intercambiada oralmente, pondo em risco a continuidade do sistema caso uma ou mais pessoas se desliguem. Manter a integridade e consistência de tais arquivos (disciplina) é difícil, fazendo com que, em ambientes de desenvolvimento utilizando técnicas manuais, acabe-se tendo somente o código fonte como informação sobre os programas. Em um ambiente assistido por computador, as informações sobre o sistema são armazenadas em um repositório mecanizado, ou seja em um banco de dados especializado. Este torna possível reconstruir as representações, mantê-las e controlar a consistência entre os diferentes documentos. Instâncias de ambiente geradas com Totem provêm suporte integrado a todos esses papéis e ferramentas.

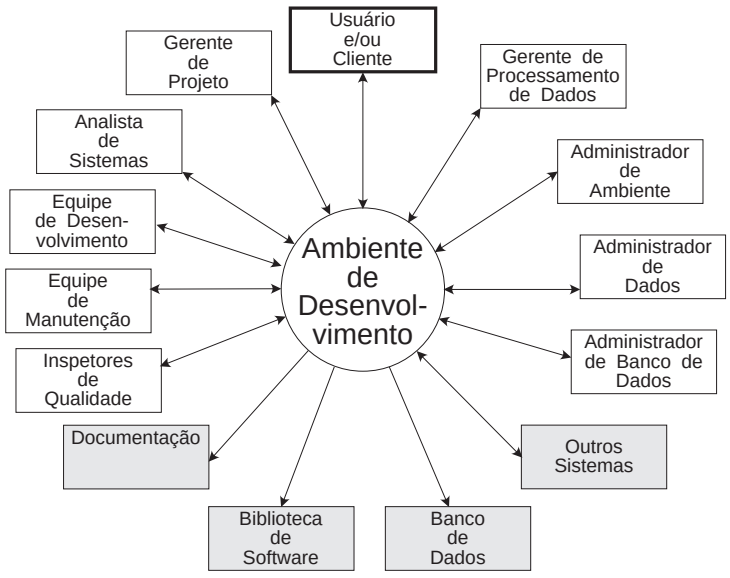


Figura 1. Papéis suportados por instâncias de ambiente geradas com Totem.

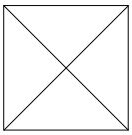

Figura 2. Ilustração de um ambiente para trabalho em equipe

A figura 2 ilustra um ambiente de desenvolvimento onde diversos *engenheiros de software* (líder de projeto, analista, projetista, programador, inspetor de qualidade) interagem através de um *repositório* para produzirem e manterem diversos *sistemas objetivo*. O repositório é formado por um conjunto de bases de dados e contém todas as informações técnicas sobre um ou mais sistemas objetivo sendo desenvolvidos ou mantidos. Ambientes necessitam, também, interagir com ferramentas externas, seja para complementar o leque de ferramentas disponíveis, seja para interagir com ferramentas que vinham sendo utilizadas.

A figura 3 ilustra a composição de uma instância de ambiente gerada por Totem. Utilizando a instância de ambiente, o engenheiro de software cria, altera e explora interativamente o conteúdo do repositório. Enquanto estiver fazendo isto, dados sobre o acompanhamento (tais como etapas concluídas, recursos consumidos, alterações realizadas, ações pendentes, medições em geral) são automaticamente coletados e armazenados na *base de acompanhamento*.

Todas as atividades técnicas são controladas pela *base de definição*. Esta base de dados descreve os processos, métodos, técnicas e linguagens de representação que podem ser utilizadas na instância de ambiente. Diferentes instâncias terão configurações diferentes, provendo suporte a atividades diferentes. Cabe ao *administrador de ambiente* determinar a configuração de um ambiente específico e cabe ao *gerente de desenvolvimento* definir a composição de cada instância. Finalmente, a seqüência de tarefas é definida e controlada pela *base de processo*. Esta determina quando devem ser utilizados que instrumentos, e quais os critérios de aceitação a serem satisfeitos antes de se poder prosseguir para uma próxima etapa. A base de processo determina ainda como o trabalho desta instância interage com o trabalho sendo realizado em outras instâncias de ambiente.

*Produtos do desenvolvimento* são os documentos de engenharia, os documentos destinados aos usuários, os relatórios de controle de evolução, os documentos de gerência, arquivos contendo texto de auxílio, as bibliotecas e os programas. Em uma instância de ambiente gerada por Totem os produtos do desenvolvimento poderão ser recriados a qualquer instante a partir das diversos repositórios e ferramentas. As bases de dados que constituem o repositório bem como as bases de definição e de processo mantém informações de controle de versão e de configuração. Pode-se, assim, controlar a evolução do próprio ambiente de desenvolvimento, bem como dos

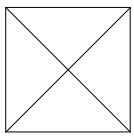

Figura 3. Composição de uma instância de ambiente

planos de desenvolvimento.

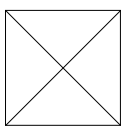

Figura 4. Bases distribuídas manipuladas por diversos desenvolvedores

Totem apoia o trabalho em equipe e a reutilização generalizada. A figura 4 ilustra o uso de Totem operando com um conjunto de *bases de software* distribuídas sendo manipuladas por diversos membros de uma equipe através de suas respectivas estações de trabalho. O repositório global de Totem consta de diversas bases de software interligadas. Estas bases de software são bancos de dados especializados que podem referenciar arquivos convencionais, tais como arquivos de código executável, bibliotecas e textos em linguagens de formatação. As bases e os arquivos podem estar em diferentes estações de trabalho, facilitando o trabalho em equipe e o reúso. O particionamento do repositório em bases de software é arbitrário, permitindo particionar segundo níveis de abstração, pessoa responsável, etapa do processo de desenvolvimento ou qualquer outro critério.

Objetos e atributos contidos nas bases de software são versionados. Desta forma, diferentes versões de um mesmo projeto poderão ser fácil e rapidamente recuperadas. As bases de software definem também como compor produtos do desenvolvimento a partir de componentes versionados. Facilitam, assim, a gerência de configuração.

## 2.2 Meta-ambiente Totem

A composição de cada instância de ambiente será definida pelo meta ambiente Totem. Este é na realidade uma instância de Totem configurada para criar e manter ambientes de desenvolvimento completos. Na figura 5 ilustramos o processo de geração e uso de instância de ambientes. O *administrador de ambiente* cria e mantém uma variedade de processos, linguagens de representação e regras de geração e de controle de qualidade. Ao iniciar um novo desenvolvimento, o administrador de ambientes seleciona uma determinada configuração desses elementos e gera o *ambiente específico*. Os critérios de configuração dependem dos requisitos de qualidade a serem assegurados, das normas de desenvolvimento em vigor na empresa e da natureza do sistema a ser desenvolvido ou mantido.

Uma vez definido o ambiente específico, o *gerente de projeto* particiona-o em várias *instâncias de ambiente*, cada qual alocada a uma estação de trabalho e utilizada por um determinado engenheiro de software. Os critérios de particionamento dependem do plano de desenvolvimento, das pessoas e dos recursos disponíveis, e depende, ainda, da natureza do sistema a desenvolver ou manter.

Utilizando as instâncias de ambiente, os *engenheiros de software* realizam o trabalho de desenvolvimento, de manutenção e de gerência, produzindo bases de software. Estas bases de software são interligadas e constituem o repositório do sistema sendo desenvolvido ou mantido.

# 3. Cenários de uso de Totem

Nesta seção apresentaremos alguns cenários de uso de Totem. Inicialmente apresentamos o uso de uma instância como uma ferramenta CASE. A seguir discutimos como utilizar instâncias para projetar e compor programas. Ao final discutimos como gerar programas a partir de especificações reutilizando uma parte significativa de especificações e projetos de fragmentos de programas.

## 3.1 Criação e manutenção de diagramas e respectivos dicionários

Esta seção ilustra o uso de Totem como uma ferramenta CASE. A figura 6 ilustra o funcionamento do conjunto de editores de Totem interagindo com um hiper-documento. Nesta figura estão sendo editadas 3 representações simultaneamente. As janelas estão apresentadas como "papel rasgado". As linhas tracejadas entre as janelas correspondem a alguns dos elos existentes entre as representações. Estes elos permitem a navegação e a validação do conteúdo das diferentes representações. Este conjunto de representações e elos formam um *hiper-documento*.

Inicialmente, seleciona-se a família de linguagens de representação a ser utilizada, por exemplo a linguagem *fluxo de dados*. Ao fazer isto, todos os meta-editores disponíveis (ver seção 4) serão automaticamente configurados para editar especificações funcionais de um sistema em conformidade com o padrão da representação de fluxo de dados adotado para os diagramas, as estruturas e as especificações. O formato e o conteúdo destas especificações são determinados pelos *programas de formulários* dos respectivos editores. Esses programas de formulários determinam o comportamento específico dos editores e podem ser alterados pelo usuário de Totem, permitindo, assim, o ajuste do ambiente às necessidades de cada equipe e projeto.

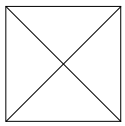

Figura 6. Ilustração de um hiper-documento

Utilizando o editor de diagramas cria-se ou mantém-se diagramas de fluxo de dados. Clicando sobre um elemento destes diagramas, ativa-se o editor de formulários deste elemento, ou o editor de estruturas centrado neste elemento. Por exemplo, ao clicar sobre o depósito de dados "Histórico escolar" pode-se selecionar a representação a examinar, que pode ser um diagrama de estrutura de dados, ou pode ser um formulário contendo a especificação detalhada dos diversos atributos que perfazem um histórico escolar, ou finalmente, pode ser um formulário ilustrando como seria a declaração da tabela correspondente ao depósito de dados.

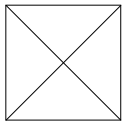

Figura 5. Fluxo de geração de instâncias de ambiente

A partir de qualquer atributo contido em uma representação sendo editada, pode-se navegar para uma janela, nova ou já ativa, contendo uma representação centrada no objeto que possui o atributo original. Alternativamente, pode-se navegar para uma janela já aberta, reposicionando o foco de atenção sobre um dos atributos exibidos na janela. Estas regras de navegação estabelecem um mecanismo de exploração e edição de hiper-documentos muito poderoso.

Ao tentar alterar um atributo em qualquer uma das janelas ativas, é verificado se este elemento está bloqueado. Caso não esteja, o elemento será bloqueado e a edição será realizada. Caso esteja bloqueado, será emitida uma mensagem. Por exemplo, o usuário pode ter alterado alguma coisa no diagrama de fluxo de dados, e agora quer alterar o mesmo atributo, só que no formulário contendo a especificação detalhada. Como resposta a esta mensagem pode-se navegar para as janelas que estão provocando o bloqueio e agir de modo a retirar os bloqueios. Por exemplo, pode-se guardar as alterações realizadas no diagrama, liberando, assim, o atributo alterado. Este mecanismo de bloqueio é necessário pois um mesmo atributo pode ocorrer repetidas vezes em uma mesma representação ou em representações diferentes, onde cada uma delas está contida em um editor ativo no momento.

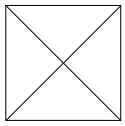

Figura 7. Composição de um ambiente de programação

## 3.2 Desenvolvimento de programas simples

Esta seção ilustra o uso de Totem para criar e manter programas. Nessa forma de uso, Totem se assemelha a modernos ambientes de programação integrados, tais como Visual C/C++, Turbo C/C++, Delphi, Visual Basic e outros. Totem somente provê mecanismos de criação e manutenção dos programas, não provê compiladores nem depuradores. Em uma instalação utilizando Totem essas últimas ferramentas serão sempre software comercial disponível no mercado.

Um programa simples redigido em C++ consta de diversas classes e funções. As funções possuem uma estrutura de algoritmo e definem uma interface. As classes agregam diversas funções, provendo as operações necessárias e suficientes para manipular uma determinada estrutura de dados. A documentação técnica de um programa está parcialmente contida no código. Por exemplo, o fato de uma função ser pública ou privada é fato a ser registrado no código. De forma, semelhante, os cabeçalhos e protótipos possuem estrutura parecida uma com a outra. É conveniente então:

- dispor de um editor sensível à estrutura dos algoritmos.
- centralizar toda a informação de modo que não se criem cópias.
- gerar os *módulos de definição* (declarações das interfaces), os *módulos de implementação* (o código fonte a ser submetido ao compilador), os *módulos de especificação* (texto de documentação a ser fornecido ao desenvolvedor que quer utilizar determinado módulo) e os *browsers* (hipertextos *on-line* tornados disponíveis ao desenvolvedor visando acelerar o desenvolvimento de aplicações a partir de componentes conhecidos). Esta geração é realizada a partir de uma base comum, evitando, assim, inconsistências e redundâncias de entrada de dados.
- gerar os textos de auxílio a serem fornecidos para os usuários dos programas.
- apoiar a manutenção e o controle da evolução do programa.

Na figura 7 o editor de diagramas de transição é utilizado para definir a dinâmica de interfaces de usuário complexas. O editor de diálogos define o aspecto de janelas de entrada ou consulta de dados. O editor de classes de interface de usuário associa código aos diálogos e interfaces de usuário. O editor de classes de armazenamento cria e mantém o código de manipulação de dados permanentes da sessão ou mesmo entre sessões. Todos estes editores interagem via o repositório e evoluem versões a medida que o programa vai sendo mantido. O controlador de configuração determina como compor um programa a partir das versões dos componentes. O gerador de *make* gera um *script* para o programa *make* de modo que o programa possa ser compilado. Finalmente, os demais geradores geram os arquivos de código fonte, documentação (entrada para algum formatador) e auxílio. A manutenção do programa é realizada pelos mesmos instrumentos. Ou seja, o código não é mantido a nível de arquivos de código fonte, e sim é mantido a nível das ferramentas utilizadas para criá-lo inicialmente.

Na realidade todos estes editores correspondem aos mesmos meta-editores utilizados em outros ambientes, somente que instanciados para apoiarem o projeto e desenvolvimento de programas.

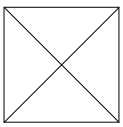

Figura 8. Organização de um ambiente de alto nível

Nesta forma simples o ambiente de programação se assemelha aos ambientes de desenvolvimento integrados. No entanto, Totem extrapola estes ambientes uma vez que permite associar referências cruzadas e código às especificações. Desta forma assegura-se a consistência entre especificações, arquitetura, projeto detalhado e código. Assegura-se também a *rastreabilidade* de especificações podendo identificar as razões para que determinada porção de código esteja presente no programa. A integração de especificações, arquitetura, projeto e código via de regra não é possível ser realizada em ambientes de programação integrados.

## 3.3 Desenvolvimento de programas a partir do reúso de especificações

Um ambiente tipo Totem torna-se mais interessante no momento em que se puder gerar código diretamente a partir de especificações.

Na figura 8 ilustramos um ambiente no qual se gera código diretamente a partir de especificações. Nesta figura, no lado *servidor*, encontra-se um banco de dados cuidadosamente definido através de seu esquema. As funções de garantia de integridade especializadas são associadas diretamente ao esquema desse banco de dados. O esquema e o correspondente código são definidos para todas as aplicações. Cada uma dessas aplicações manipula uma determinada visão da base de dados. Uma vez identificada a visão da base de dados a ser manipulada pela aplicação, pode-se gerar o processador servidor. Note que ao fazer isto pode ser automaticamente incluído todo o código necessário para manter a correta interação entre estações cliente e servidor. Esta forma de gerar servidores estabelece um elevado grau de reúso.

Do lado *cliente*, o usuário cria os diálogos e associa os elementos ao esquema da base de dados. Ao fazer isto, está-se definido a visão da base de dados que será manipulada. Note que esta visão pode estar delimitada a priori, impedindo assim o usuário de associar itens de diálogo a elementos da base de dados para as quais não tem autorização de acesso. É sabido que a correta manutenção de uma base de dados, requer a atualização de elementos que estão fora da visão permitida a um determinado usuário. Como os programas são gerados, este problema é contornado através do uso do esquema completo para a geração do programa e a permissão de acesso limitado ao definir a aplicação.

# 4. Esboço do projeto

Nesta seção apresentamos um esboço da arquitetura de Totem. Este esboço tem por finalidade evidenciar a viabilidade do projeto.

## 4.1 Experiência prévia

Em 93 foi concluído o desenvolvimento de Talisman que, em última análise, é um protótipo de Totem. Este protótipo é mono-base e não possui controle de versão. Talisman satisfaz quase integralmente os demais requisitos identificados acima, tais como: pode ser utilizado, com restrições, como um meta-ambiente, permite a instanciação de ambientes específicos, e utiliza uma base de definição para definir a semântica da base de software. Talisman tem sido utilizado com sucesso no desenvolvimento de sistemas e em pesquisa envolvendo métodos e linguagens de representação. Em particular Talisman tem sido utilizado para desenvolver a si próprio. Além disso, estão sendo desenvolvidas diversas ferramentas em Talisman que têm por objetivo

facilitar o desenvolvimento de Totem. Detalhes sobre Talisman podem ser encontradas em [STAA93].

## 4.2 Aspectos de arquitetura

Totem opera com *famílias de linguagens de representação*. Cada família de linguagens de representação define um conjunto de *linguagens de representação elementares*. Cada linguagem elementar possui um editor próprio. Este editor é, na realidade um meta-editor instanciado para editar a linguagem de representação em questão. A figura 9 ilustra a interação dos diversos meta-editores e mostra como estes são especializados pelo conteúdo da base de definição. São exemplos de meta-editores:

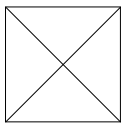

Figura 9. Arquitetura de meta-editores

- *editor de formulários*, utilizado para editar texto composto a partir de diversos atributos contidos nas bases de software. É utilizado, por exemplo, para editar: especificações contidas em dicionários de dados, código de programa composto a partir do conteúdo das bases de software e documentação extraída de especificações contidas nas bases de software.
- *editor de estrutura*, utilizado para editar estruturas arborescentes definidas por relações selecionadas. É utilizado, por exemplo, para editar estruturas de algoritmos, estruturas de tarefas, estruturas de re-escrita de métodos, estruturas de menus, e estruturas de documentos.
- *editor de diagramas*, utilizado para editar diagramas de especificação e projeto de sistemas. É utilizado, por exemplo, para editar diagramas de fluxo de dados, diagramas de entidade e relacionamento e "*state charts*".
- *editor de diálogos*, utilizado para editar a composição, o formato de diálogos e a associação dos elementos de diálogos ao código executável (não consta da figura).

Esta arquitetura permite adicionar-se novos meta-editores a medida que estes se fizerem necessários. Ao criar um novo meta-editor os programas que formam Totem e os esquemas de bancos de dados utilizados precisam ser modificados. Espera-se que a adição de novos meta-editores se dê lentamente.

Ao desenvolver um conjunto de ferramentas, desde que estas não requeiram meta-editores especializados, basta instanciar os meta-editores disponíveis. Como foi visto na seção 2, esta instanciação é realizada no nível do meta-ambiente, utilizando linguagens especializadas para a criação e manutenção de ferramentas.

As regras de seleção e formatação dos atributos que compõem as diferentes representações são definidas por programas e esquemas (*programas de formulários*) contidos na base de definição. Programas de formulários são fornecidos junto com Totem, podendo ser alterados e expandidos pelo usuário de Totem. A linguagem de programação de formulários está sendo projetada e desenvolvida considerando as necessidade de um meta-ambiente de engenharia de software. Todos os meta-editores interagem através de uma base de trabalho que estabelece um padrão de comunicação interno (*via de software*). Esta arquitetura permite a adição de novos meta-editores à medida que se tornem necessários.

Na figura 10 apresentamos a arquitetura de um meta-editor. Todos os meta-editores têm uma arquitetura semelhante a esta. Os elementos contidos em retângulos correspondem a programas

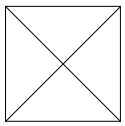

Figura 10. Organização de um meta-editor

dirigidos por tabelas (interpretadores). As tabelas utilizadas pelos programas estão identificadas nas elipses. As arestas informam o fluxo de dados entre tabelas e programas.

O conjunto de tabelas é armazenada na base de definição. Estas tabelas são armazenadas em forma otimizada para a execução. Através de um compilador, o administrador de ambientes transforma programas em linguagem fonte para programas redigidos nestas tabelas. O esquema da base de definição otimiza a procura e o acesso aos elementos das diversas tabelas. Consegue-se assim um desempenho satisfatório para os diferentes editores. Conforme discutido na seção 2, as bases de definição são geradas a partir da base de software contida no meta-ambiente. Ou seja, um dos componentes de Totem é um gerador de bases de definição capaz de converter o conteúdo das bases do meta-ambiente (código fonte) nas tabelas armazenadas nas bases de definição (código objeto). Esta arquitetura de editores foi utilizada em Talisman e tem se mostrado satisfatória mesmo em máquinas de baixo desempenho.

## 5. Epílogo

Neste documento apresentamos alguns cenários de uso de Totem. Inicialmente mostramos como Totem pode ser utilizado para gerar ambientes específicos e instâncias de ambientes. A seguir mostramos exemplos de uso de Totem variando de um uso bem simples como uma ferramenta CASE até o uso como um gerador de programas a partir de representações de alto nível de abstração. Concluímos com um esboço da arquitetura de Totem.

O objetivo deste documento é facilitar a compreensão de aspectos operacionais de Totem. Não almejamos com este documento estabelecer um confronto entre os objetivos de Totem e de outros ambientes de desenvolvimento. Esta análise comparativa é objetivo de outros documentos, em particular:

## Referências bibliográficas


[STAA93] *Ambiente de Desenvolvimento de Software Talisman, Manual do Usuário*; Staa Informática Ltda.; Rio de Janeiro; RJ; 1993.

[Staa95] Staa, A.v.; Cowan, D.D.; *An Overview of the Totem Software Engineering Meta-Environment*; Monografias em Ciência da Computação MCC.Inf.PUC-Rio 35/95, Departamento de Informática, PUC-Rio; 1995

[Staa96] Staa, A.v.; *Totem: Repository Schemas*; Monografias em Ciência da Computação, Departamento de Informática, PUC-Rio; em preparação