CONTRAM: MIDDLEWARE PARA INTEROPERABILIDADE DE REDES HETEROGÊNEAS DE CONTROLADORES SEMAFÓRICOS EM SISTEMAS DE TRANSPORTES INTELIGENTES.

Lincoln Luiz de Morais
Instituto de Informática/Universidade Federal do Rio Grande do Sul
Porto Alegre, RS, Brasil, lincoln@inf.ufrgs.br

Cláudio Fernando Resin Geyer
Instituto de Informática/Universidade Federal do Rio Grande do Sul
Porto Alegre, RS, Brasil, geyer@inf.ufrgs.br

Abstract

Systems that use computational technologies in treatment the problems relative to the transit are classified as ITS or Intelligent Transportation System. Urban Traffic Systems Management manages flow of vehicles and road occupation using traffic control devices such as traffic lights and sensors and its respective controllers. Computationally, are relevance factors: the interoperability between these controllers and standardizations adopted. A urban traffic control global system (hardware and software) normally is implemented by stages, being acquired controllers of different manufacturers and models, making it difficult the integration due its proprietary technologies.

This work presents part of CONTRAM, a middleware that, treating the traffic controllers installed in road mesh based in distributed systems and computer networks paradigms, it can be used as interface between traffic management applications and control devices controllers, allowing the integration of different specifications of controllers in an only system. This work treats only the computational architecture about integration.

Keywords: ITS, middleware, CONTRAM, traffic control, SNMP, distributed systems.

Resumo

Sistemas que utilizam tecnologias computacionais no tratamento de problemas relativos ao trânsito são classificados como *ITS* ou *Intelligent Transportation System*. Sistemas de Gerenciamento de Tráfego Urbano gerenciam o fluxo de veículos e a ocupação da malha viária utilizando dispositivos de controle de tráfego como semáforos e sensores e seus respectivos controladores. Computacionalmente, são fatores relevantes: a interoperabilidade entre estes controladores e padronizações adotadas. Um sistema global (*hardware* e *software*) de controle de tráfego urbano normalmente é implementada por etapas, sendo adquiridos controladores de diferentes fabricantes e modelos, dificultando a integração entre os mesmos em função de suas tecnologias proprietárias.

Este trabalho apresenta parte do *CONTRAM*, um *middleware* que, tratando os controladores de tráfego instalados ao longo da malha viária baseado nos paradigmas de sistemas distribuídos e redes de computadores, possa ser utilizado como interface entre as aplicações computacionais de gerenciamento de tráfego e os controladores de dispositivos de controle, permitindo a integração de diferentes especificações de controladores em um único sistema. Os aspectos tratados neste trabalho dizem respeito apenas à integração do *CONTRAM* com os controladores de dispositivos de controle.

Palavras chaves: *ITS*, *Intelligent Transportation Systems*, *Middleware*, *CONTRAM*, Controle de Tráfego Urbano, *SNMP*, Sistemas Distribuídos.

1. Introdução

A Engenharia de Tráfego representa o ramo da Engenharia de Transportes que se ocupa do planejamento, projeto geométrico, operações de tráfego em redes viárias e terminais, assim como o relacionamento com outros modos de transporte [5]. Associado à falta de planejamento urbano efetivo no desenvolvimento industrial e/ou comercial e, conseqüentemente, na expansão territorial e populacional de importantes cidades ao redor do mundo, surgiram vários problemas de impacto social, dentre eles, o da ocupação da malha viária e os problemas específicos decorrentes dessa ocupação, tais como congestionamentos, atropelamentos e assaltos no trânsito, com ou sem vítimas. Obras civis como construções de viadutos, de novas vias ou alargamento das já existentes ou ainda melhorias no sistema de transportes coletivos, podem contribuir positivamente para minimizar os problemas acima citados. Outras alternativas, a um custo inferior, surgem a partir ou da criatividade humana, como é o caso do sistema de rodízio adotado pela cidade de São Paulo, ou a partir da utilização de tecnologia computacional no controle e gerenciamento do tráfego urbano, no qual se incluem os sistemas que buscam minimizar congestionamentos urbanos e suas conseqüências, originando termos como *CATE* (*Computer Aided Traffic Engineering)* e *ITS*.

Buscando atingir os objetivos propostos, os SGTU's ou Sistemas de Gerenciamento de Tráfego Urbano atuam coordenando e sincronizando os tempos semafóricos de um grupo de semáforos, através de seus controladores, em função das necessidades da demanda, detectadas através de dados obtidos a partir de outro tipo de dispositivo de controle, os sensores, localizados ao longo das vias e atuando como monitores das condições do tráfego.

A utilização de tecnologia inteligente, baseada em microprocessadores, nos controladores atuais permite um escopo maior de funções dentro da atividade de controle e monitoramento, porém, muitas destas facilidades não são suportadas pelos sistemas mais antigos, ou até mesmo de uma geração anterior, devido a falta de um padrão aberto que possibilite a interoperabilidade, já que cada fabricante adota soluções proprietárias. Para a captura automática, e realisticamente mensurados, dos dados de entrada sobre os quais o SGTU realiza operações para a tomada de decisão, faz-se necessário a integração deste com os controladores e entre os próprios controladores, valendo-se de todas as facilidades tecnológicas na obtenção dos dados. Uma ferramenta de controle de tráfego urbano normalmente é implementada por etapas, em que diferentes especificações de controladores são adquiridos por motivos diversos. Tais diferenças freqüentemente geram incompatibilidades nas trocas de dados. Portanto é interessante buscar uma forma de viabilizar a comunicação entre todos esses equipamentos, de forma que investimentos realizados possam ser preservados através do estabelecimento de um formato padrão para a troca de dados entre os mesmos, permitindo o rompimento com a dependência atual causada pelo uso de sistemas proprietários sem contudo desprezar o conhecimento acumulado. A principal crítica por parte dos usuários tem sido a inflexibilidade dos sistemas disponíveis comercialmente, muitos são tidos como "caixas-pretas" [5].

Este trabalho buscou nos paradigmas de sistemas distribuídos e de redes de computadores, uma alternativa ao problema da interoperabilidade, apresentando um modelo de rede de controladores voltado ao apoio à gestão e operacionalização do tráfego urbano, que possa ser utilizado como interface entre um SGTU e diferentes modelos e tipos de controladores em um único sistema, permitindo que controladores com ou sem inovações tecnológicas possam ser adicionados e/ou removidos, de forma transparente à aplicação de gerenciamento, ou seja, sem que esta sofra alterações em seu código fonte, promovendo sua revitalização e adequação de acordo com necessidades específicas.

O escopo do modelo é prover mecanismos, mapeamento de recursos, comunicação e conversão de dados, que permitam atender às operações, consulta e configuração dos valores das variáveis utilizadas para controlar, monitorar e gerenciar o fluxo de veículos, solicitadas pelo SGTU, liberando-o de conhecer detalhes técnicos tanto dos controladores como dos envolvidos na conversão e troca de dados. Está fora do escopo interpretar as operações solicitadas pelo SGTU. O modelo foi batizado de ***CONTRAM***, um acrônimo de ***CON****trollers* ***TRA****ffic* ***M****iddleware*. Neste trabalho será apresentado apenas o modelo da solução adotada para integrar os controladores ao *CONTRAM*, embora o modelo originalmente proposto baseie-se em uma arquitetura multicamadas, 4-*tier*, podendo atuar em uma configuração centralizada ou distribuída, dependente das características do SGTU, trocando dados através da Internet, e projetado tendo em mente a utilização de padrões abertos da indústria da Informática e de Transportes, a interoperabilidade entre diferentes elementos tecnológicos de controle de tráfego e a expansibilidade de um SGTU.

Este trabalho é composto por 4 seções. A próxima seção trata de conceitos básicos do tráfego urbano e seu controle; em seguida são apresentadas as características do *CONTRAM* para a integração dos controladores e por fim as conclusões.

2. Tráfego urbano e seu controle

Genericamente, um sistema de tráfego é composto por vias, interseções e pelo movimento de elementos circulantes, veículos e pedestres, que são os usuários desse sistema. O movimento dos veículos originam as chamadas correntes de tráfego ou fluxo.

Dispositivos de controle como os semáforos buscam otimizar o fluxo através do sincronismo dos tempos semafóricos ou tempo de duração dos intervalos de verde e vermelho. O conjunto de tempos semafóricos é chamado de *plano semafórico*, que pode ser concebido dinamicamente em função da demanda momentânea ou pré-programado com base em dados históricos do comportamento do tráfego em dias e horários específicos. Estes planos são ativados em função da variação do comportamento do fluxo, em diferentes horários do dia e dias da semana segundo uma tabela de horários de ativação, ou ainda em condições específicas, como por exemplo, uma manifestação pública e detecção de falha no equipamento.

A Figura 2.1 exemplifica uma interseção ou cruzamento entre uma avenida de via dupla e uma rua de via simples. Por grupo entende-se um conjunto de semáforos que possuem a mesma fase ou intervalo em determinado instante. No exemplo são tratados dois grupos semafóricos para veículos, indicados por G1 e G2, e três grupos para os pedestres, indicados por P1, P2 e P3. A Figura 2.2 mostra um exemplo de um possível plano semafórico simplificado para esta interseção.

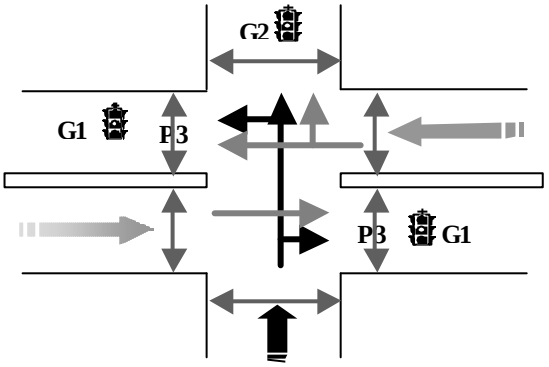

FIGURA 2.1 - Exemplo de uma interseção, com indicadores de sentido das vias e movimentos, para pedestres e veículos, e grupos semafóricos para veículos (G1 e G2) e pedestres (P1, P2 e P3).

		Seqüência em que as lâmpadas acendem (evento paralelo para diferentes fases (F1 à F5))							
Grupo	**Fase**	1	2	3	4	5	6	7	8
G1	F1	V	V	A	R	R	R	R	R
G2	F2	R	R	R	R	V	A	R	R
P1	F3	V	r	R	R	R	R	V	V
P2	F4	R	R	R	R	V	V	V	r
P3	F5	R	R	R	R	R	R	V	r
		16	1	1	1	8	1	3	3
		Tempo em que as lâmpadas permanecem acessas ao longo da Fase.							

FIGURA 2.2 – Exemplo de plano semafórico simplificado para dada interseção, onde o ciclo se inicia em verde (*V*), passando pelo amarelo (*A*), vermelho (*R*) e vermelho de pedestre (*r*).

Os semáforos funcionam conectados e recebendo dados de um equipamento eletrônico microprocessado conhecido por controlador de dispositivos de controle, ou simplesmente, controlador. Este equipamento possui recursos voltados à comunicação de dados, à realização de um pequeno volume de processamento e ao armazenamento dos planos semafóricos e tabelas diversas, estando todo o seu funcionamento baseado nos planos semafóricos, cujos tempos das fases dimensionados nos mesmos são os principais parâmetros de entrada ao processamento a ser executado. Estes planos semafóricos podem ser programados através de um equipamento específico conhecido como *Programador de Controlador* ou através da aplicação que gerencia o tráfego, permanecendo armazenados na memória *RAM* do controlador quando estiver habilitado ou for o plano semafórico vigente, e em memória do tipo *EPROM*, também no controlador, quando estiver desabilitado. Outra alternativa é armazenar estes planos em um microcomputador que executa o SGTU e está conectado ao controlador a ser atuado.

Segundo [8] e [3], os semáforos podem ser classificados em função do seu modo de operação ou estratégia operacional, que é definido pelo controlador ao qual o mesmo está conectado. Um destes modos operacionais é conhecido como *modo central*, ou seja, quando o plano semafórico a ser ativado é escolhido pela central de controle através de um SGTU.

Quando um controlador está conectado a uma rede, diz-se que o mesmo está operando na modalidade *Coordenado* [8], ou seja, está obrigado a respeitar uma defasagem de tempo no início de cada ciclo, garantindo, por exemplo, a chamada *onda-verde*. Uma rede poderá conter um número máximo de controladores, sendo um deles designado para gerar o sincronismo da rede. Este é o gerente de comunicação ou referencial, com capacidade de consultar e alterar a programação de qualquer outro controlador sob sua coordenação. O comando de *forçamento remoto* faz com que qualquer controlador da rede, ou todos, funcionem de maneira diferente do programado no plano semafórico durante um período de tempo, permitindo que a rede se comporte de acordo com uma necessidade momentânea do controlador referencial. Fisicamente, os controladores são interligados por uma interface serial, numa configuração multiponto, através de cabos metálicos, permitindo comunicações confiáveis até uma distância máxima de 3500 mts [3] e 5000 mts [8].

3. *CONTRAM*

[4] define como arquitetura de sistema um modelo que expressa as funcionalidades do sistema e a integração entre estas funcionalidades em um alto nível de abstração, podendo existir diversas sub-arquiteturas ou camadas para que o mesmo seja abrangente, hoje e no futuro. As finalidades básicas da elaboração de uma arquitetura de sistema são: a) Construir sistemas integrados eficientemente; b) Garantir a expansibilidade do sistema; c) Promover padrões. Dada a abrangência dos sistemas para *ITS* é de fundamental importância a elaboração de uma arquitetura de sistema, por envolver diversas áreas sociais que são suscetíveis a mudanças comportamentais.

3.1 Arquitetura funcional do *CONTRAM*

A arquitetura do *CONTRAM* foi estruturada em sub-arquiteturas, que por suas vez foram subdivididas em camadas onde estão definidas as funcionalidades do mesmo. O objetivo da especificação e implementação de uma arquitetura multicamadas é garantir a coesão, a modularidade e a expansibilidade do sistema como um todo, uma vez que para agregar novas funcionalidades ou rever as já existentes, modifica-se a camada que fornece essa funcionalidade, seja ela uma nova tecnologia ou um novo serviço, permitindo, do ponto de vista comercial, a criação de novas oportunidades de negócios e mercados. Basicamente as camadas apresentadas pelo modelo são: a) *Apresentação*; b) *Regras de Negócios*; c) *Infra-estrutura*; d) *Dados*, integradas conforme Figura 3.1

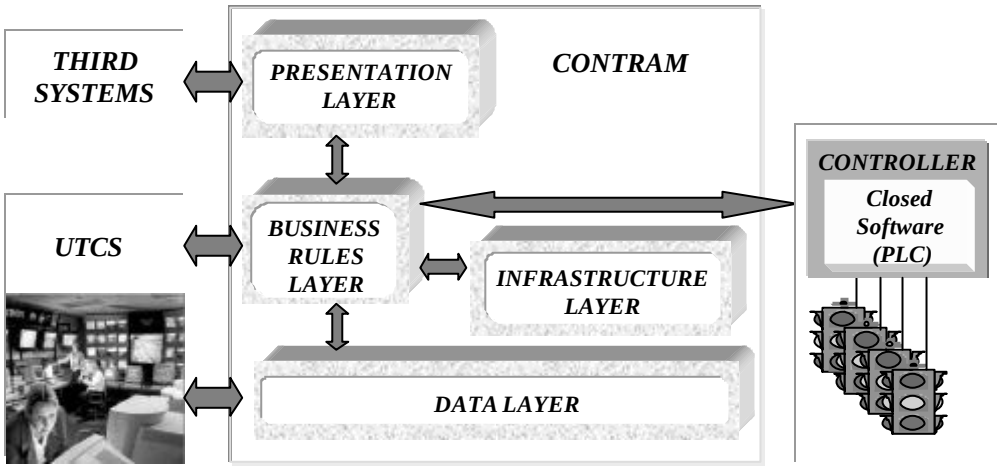

FIGURA 3.1 – Integração entre as múltiplas camadas do *CONTRAM*, onde as setas indicam as interações entre as camadas e sistemas (SGTU e Controladores).

Tipicamente, os dados em uma aplicação de *ITS* são gerados por diferentes tipos de sensores geograficamente distribuídos ao longo de uma área de controle de tráfego, também chamada de CTA, que mensuram diferentes parâmetros da aplicação, cada um fornecendo um panorama da situação momentânea. A infra-estrutura de comunicação e tecnologias de dispositivos de controle de tráfego estão permitindo projetos de aplicações que funcionem em ambiente de computação distribuída, combinando os dados obtidos para oferecer diversos serviços aos clientes, também distribuídos [1]. O modelo *CONTRAM* assume a cidade dividida em CTA's, cada qual com uma rede de controladores e gerenciada por um computador regional ou *Inspector Computer* (*IC*), que interage com o computador do centro operacional ou *Delegate Computer* (*DC*) onde está instalado o SGTU.

Com relação ao empacotamento de *software*, apenas conceitualmente, foram projetados dois módulos logicamente equivalentes, o *Central Module* (*CM*) que é instalado no *Delegate Computer* atuando como interface com o SGTU e o *Remote Module* (*RM*) instalado nos computadores que estão distribuídos ao longo da malha viária, ou *Inspector Computer*, atuando como interface com os controladores. Os serviços previstos podem ser ativados ou não em função do local de execução. Ambos os módulos acessam bases de dados que armazenam os dados manipulados pelo *CONTRAM*. Esses dados podem conter valores resultantes de processamento realizado ou parametrizações utilizadas pelo modelo. A visão global dos dados do sistema é mantida no *DC* e as visões parciais, referentes aos controladores das CTA's administradas, são mantidas no *IC*. O domínio do *CONTRAM* limita-se ao atendimento das solicitações de operações de atuação sobre os controladores solicitadas pelo SGTU, não se responsabilizando por quaisquer decisões tomadas a nível de gerenciamento. Atendidas as normas pré-definidas pelo modelo para a integração, toda e qualquer solicitação de operação de consulta ou configuração recebida de um sistema de controle será atendida, não sendo avaliado o seu impacto no comportamento do tráfego urbano. A Figura 3.2, contextualiza o ambiente controle de tráfego como um todo, demonstrando a arquitetura funcional do *CONTRAM* em alto nível e a delimitação do seu escopo .

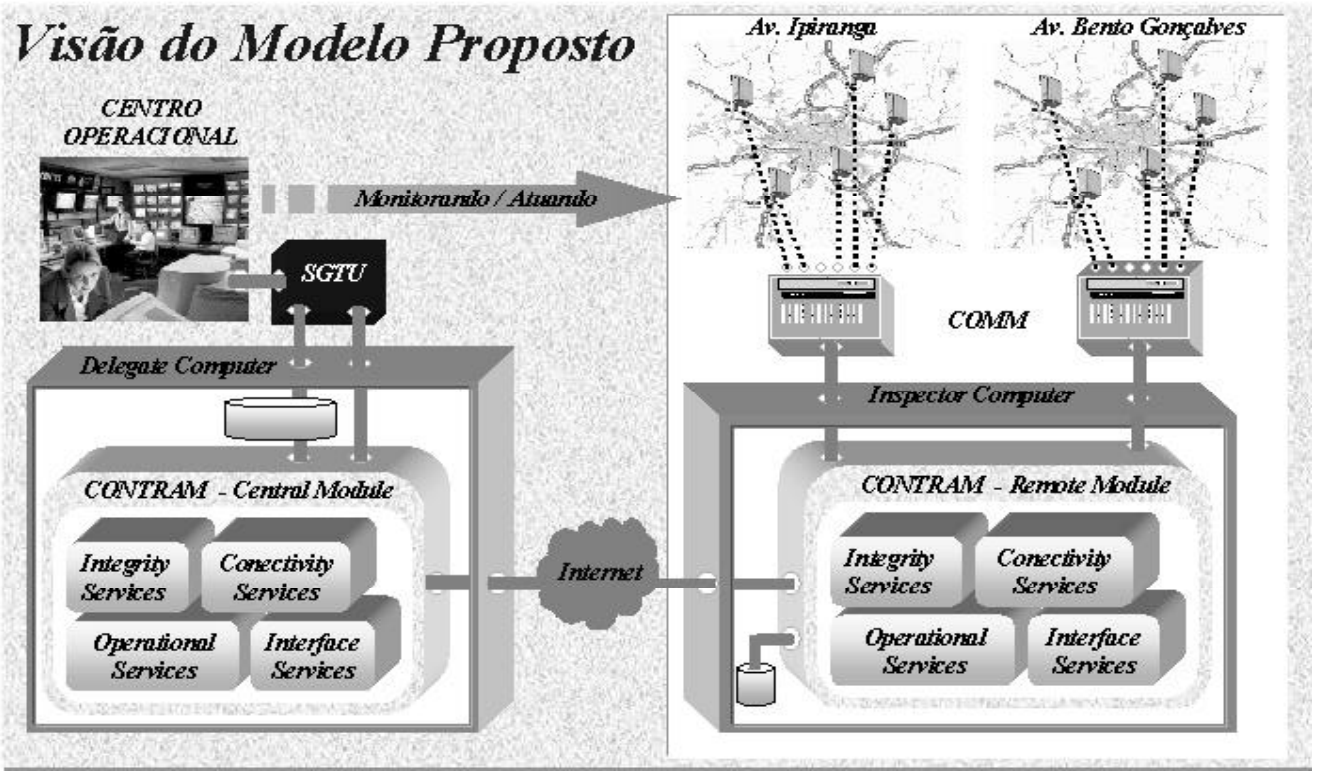

FIGURA 3.2 – Contexto de controle de tráfego, demonstrando a arquitetura funcional e a delimitação do escopo do *CONTRAM*.

Dado o objetivo do *CONTRAM*, são definidas duas interfaces com características distintas de concepção e funcionamento, uma com os sistemas de alto-nível ou os SGTU's e outra com os sistemas de baixo-nível ou os controladores de dispositivos de controle de tráfego, sendo o escopo deste trabalho limitado a esta última.

3.2 Integração *CONTRAM* - Controladores

As funcionalidades providas pelo modelo são chamado de serviços, que englobam os subserviços. Dentre os serviços providos pelo modelo existem aqueles que são chamados de *nativos*, constituindo-se na base de funcionamento do mesmo e os *agregados*, que dão funcionalidades extras podendo ser alterados de forma que o funcionamento básico não seja alterado. Esses serviços foram agrupados hierarquicamente baseados em sua abrangência funcional dentro do funcionamento do *CONTRAM*, conforme Figura 3.3.

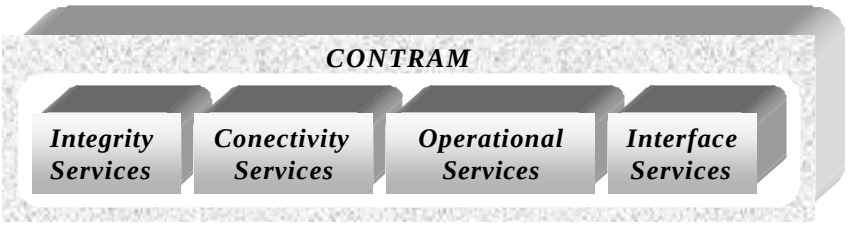

FIGURA 3.3 – Hierarquia dos serviços previstos pelo *CONTRAM*.

Para a integração do *CONTRAM* com os controladores são utilizados subserviços de *Interface Services* e *Conectivity Services*, voltados respectivamente ao interfaceamento com os sistemas computacionais a serem integrados e à transferência de dados. Os serviços de conectividade são utilizados em duas situações distintas, onde uma delas é para permitir a troca de mensagens entre o *CONTRAM*, através de um módulo de *software* com mecanismos de gerente, e os controladores, através de módulos de *software* com mecanismos de agente que convertem as mensagens recebidas do módulo gerente para serem processadas. Nesse caso é utilizado o protocolo *SNMP*, coexistindo as figuras do gerente e agente *SNMP* [2], [6]. De forma a preservar a arquitetura de *hardware* dos controladores, optou-se por trabalhar com agentes do tipo *proxy* e instalados no *IC* ao qual o controlador referencial está conectado.

A ferramenta que está sendo utilizada para o desenvolvimento dos serviços de integração entre o *CONTRAM* e os controladores, o *Java Dynamic Management Kit* ou *JDMK* [7], oferece algumas funcionalidades adicionais que foram incorporadas ao modelo proposto. Dentre elas está o conceito de *cascading* ou hierarquia de agentes, onde há um *Master Agent* que distribui os serviços de supervisão para um grupo de agentes subalternos, os *Leaf-node Agents*, gerenciando-os. Esses *Leaf-node Agents* são componentes administráveis chamados de *Mbeans* e responsáveis pela interação com os recursos a serem gerenciados, neste caso os controladores. Esses *Mbeans* são os próprios agentes-*proxy*, podendo ser integrados, retirados ou atualizados em tempo real, sem a modificação de outros componentes do sistema como um todo. Um *Master Agent* suporta vários agentes-*proxy*. A Figura 3.4 ilustra o funcionamento da tecnologia.

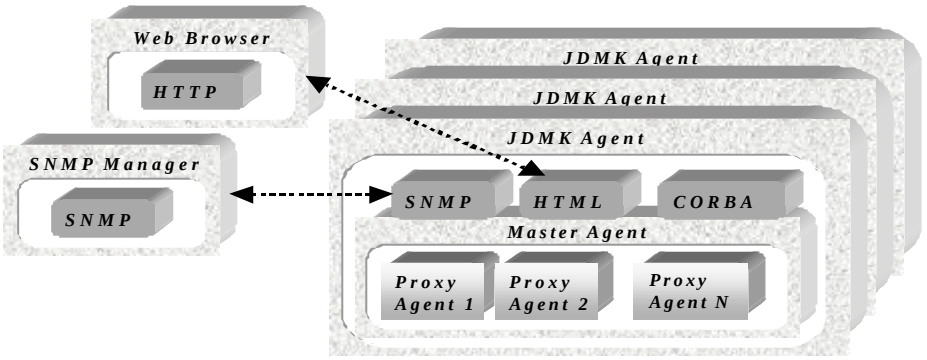

FIGURA 3.4 – Estrutura hierarquica de agentes dinâmicos, que define um "ambiente" agente (*JDMK Agent*) que contém adaptadores de protocolos, um agente principal (*Master Agent*) e agentes secundários (*Proxy Agent*).

Cada controlador possui o seu respectivo agente-*proxy* que permanece ativo o tempo todo trocando mensagens *SNMP* com um módulo gerente, nativo do *CONTRAM*. Esse agente-*proxy* atua como uma *API* do controlador, conforme Figura 3.5, à frente.

Considerando as possíveis formas de distribuição de recursos permitidos pelo *CONTRAM*, que os agentes-*proxy* serão referenciados por um mecanismo nativo do modelo, o gerente *SNMP*, e coexistem vários em um mesmo *IC*, foi adotada uma padronização quanto nomenclatura e localização dos mesmos, ambas definidas no instante de sua instalação, visando organização e controle sobre o ambiente operacional. O nome do agente-*proxy* deve ser o mesmo utilizado para identificar o controlador ao qual irá converter as mensagens, que é único em todo o sistema, e a sua localização deve estar associada a um repositório dentro da estrutura de repositórios criada pelo usuário administrador, juntamente como os demais dados por ele manipulados. O *CONTRAM* classifica um controlador como um usuário direto, sendo necessário o seu cadastramento e de seus parâmetros, permitindo sua identificação, localização e efetiva integração ao sistema como um todo. Integração esta que dá-se através do método *plug in* com ligação dinâmica, de maneira tal que um agente-*proxy* pode evoluir em suas funcionalidades refletindo a própria evolução do controlador. Quando se fizer necessário a sua atualização, deve-se desativa-lo e sobrepor o seu

conteúdo mantendo-se os mesmos parâmetros relativos à localização e identificação, de forma que quando o mesmo for ativado, o *Master Agent* e o gerente *SNMP* passam a interagir com esse novo agente-*proxy* suportando as novas funcionalidades, conforme Figura 3.4. Com isso mantém-se a modularidade do sistema como um todo.

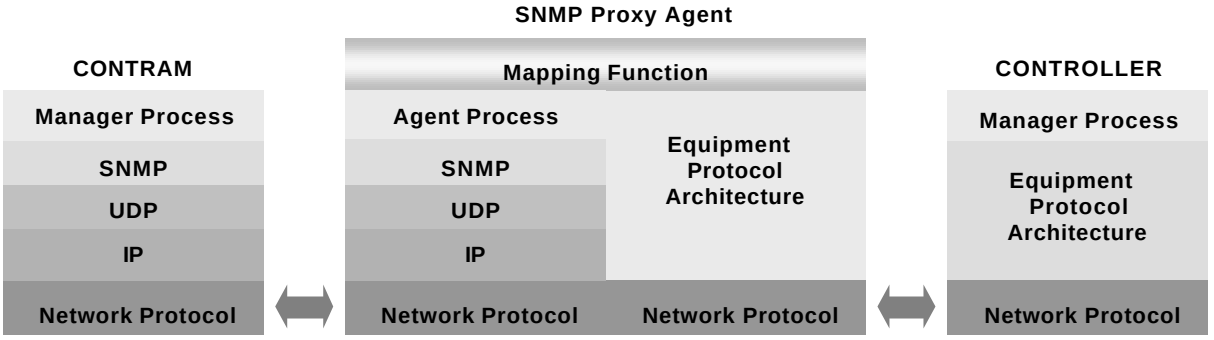

FIGURA 3.5 – Estrutura interna de um agente-*proxy SNMP*.

Quanto à concepção desse agente-*proxy*, de inteira responsabilidade do fornecedor do controlador, sugere-se que seja possível configurar parâmetros com relação a identificação e localização de recursos com o qual o mesmo interage, como as *MIB*'s, os controladores e o gerente *SNMP*, permitindo eventuais reconfigurações do ambiente, devido à critérios organizacionais ou em ocorrência de falhas.

As Figuras 3.6 e 3.7 demostram, respectivamente, o diagrama de seqüência simplificado para uma operação de obtenção do valor de um dado do controlador e o diagrama conceitual da troca de dados na interface específica.

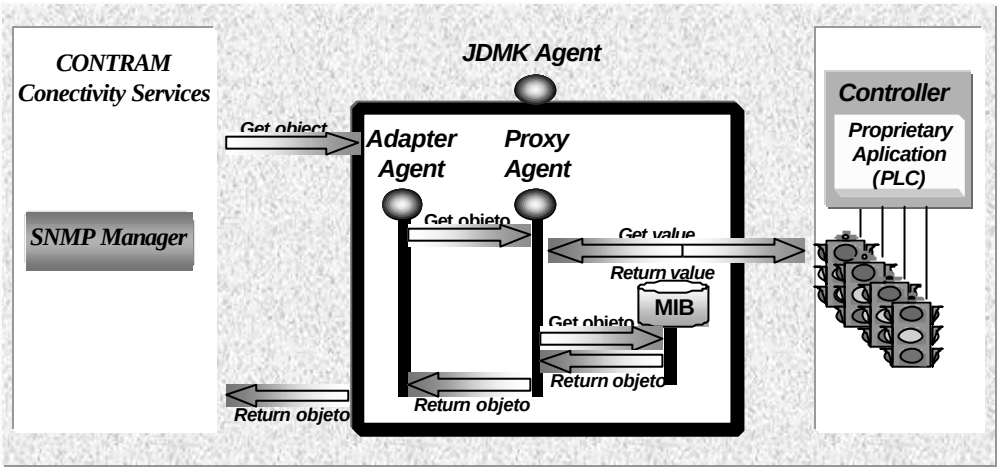

FIGURA 3.6 – Diagrama de seqüência de operações de obtenção de um valor junto ao controlador.

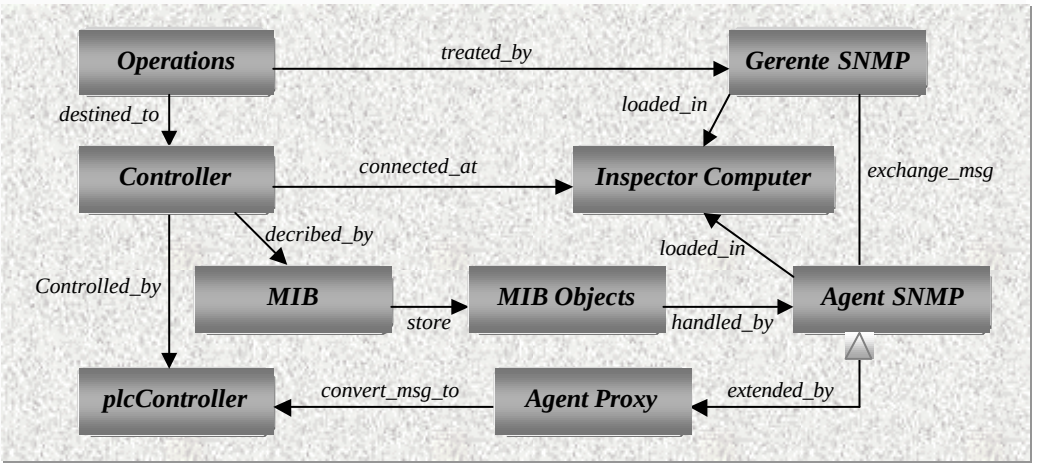

FIGURA 3.7 – Diagrama conceitual de uma troca de dados envolvendo *CONTRAM* e os controladores.

3.3 Organização dos recursos gerenciáveis nos *IC*

A implementação do agente para atuar sobre o controlador varia de acordo com o modelo e arquitetura deste, sendo identificadas algumas alternativas para tal. São elas: *a)* inserir um agente *SNMP* no módulo de controle do controlador. Considerando que o mesmo já possui suporte aos protocolos da pilha *TCP/IP*, o acréscimo de um agente *SNMP* e sua *MIB* poderia causar algum impacto no ciclo de execução das instruções. Entretanto, no caso de não apresentar suporte ao *TCP/IP*, o custo de projeto e implementação seria, provavelmente, alto. Outro agravante seria em relação a uma provável alteração na sua arquitetura de *hardware*, considerando a inclusão do agente, *MIB* e a pilha de protocolos; *b)* implementar o agente *SNMP* no módulo de comunicação. Essa alternativa minimiza o impacto na arquitetura do sistema como um todo, já que somente um módulo auxiliar do controlador foi alterado. Nesse caso, o módulo de comunicação deve dar suporte, além da pilha de protocolo *TCP/IP*, ao agente-*proxy*, já que o *SNMP* não é nativo; *c)* outra alternativa, que foi adotada pelo *CONTRAM*, é implementar o agente em um equipamento de uso genérico, como é um microcomputador, permitindo um escopo maior na função supervisória. Uma das vantagens dessa alternativa é que fica preservada a arquitetura do controlador, estando a sua *MIB* e o seu agente-*proxy* instalado no microcomputador ao qual o mesmo esta conectado. O benefício direto dessa alternativa é a menor relação custo x benefício para a integração do controlador com o mundo *SNMP*, colocando toda a dificuldade dessa integração sob a responsabilidade do *CONTRAM*. A Figura 3.8 ilustra essa alternativa de implementação.

FIGURA 3.8 – Agente *SNMP proxy* localizado em *IC* ao qual o controlador está conectado.

Cada controlador referencial possui um único agente-*proxy*, instado no *IC* ao qual o controlador está conectado, sendo utilizando o agente gerente *SNMP* configurado no *remote model* para trocar dados com o SGTU.

O fato do *CONTRAM* trabalhar com localização de recursos parametrizada permite uma grande flexibilidade quanto à distribuição dos dados. Um aspecto relativamente importante que pode causar problemas de inconsistências, dada essa liberdade de nomenclatura e localização, é a falta de critérios, padronizações e gerenciamento na escolha destes parâmetros, como nomes e endereços. Logo é importante observar critérios de nomenclatura e localizações.

Este trabalho sugere uma padronização de nomenclatura e localização para as *MIB*'s, base de dados, controladores, agentes-*proxy* e agentes-*adapter*, que são os recursos gerenciáveis com um maior número de ocorrências no modelo, visando organização e controle sobre o ambiente operacional, agregando aos mesmos algumas informações que facilitam a sua identificação em função do tipo, conteúdo e localização lógica e física de instalação. Deve ser salientado que o administrador tem a liberdade de atribuir nomes e criar uma estrutura organizacional da forma que lhe for mais conveniente dentro do seu modelo de malha viária ou distribuição física dos recursos; tanto nome como localização devem ser inseridos na arquitetura do *CONTRAM* no instante da instalação do recurso.

Quanto à nomenclatura, é sugerido um prefixo identificador e um sufixo quantificador. O prefixo é indicado para identificar o tipo de recurso, conforme Tabela 1, e o sufixo para diferenciar um único recurso dentre os vários similares, devendo ser um algarismo alfanumérico com duas casas, com valores variando entre 00 e ZZ e indicando uma ordem de precedência. Tanto o nome do *DC* ou *IC* é definido pelo usuário administrador. Para compor os nomes das bases de dados são utilizados o nome do computador, *inspector ou delegate*, no qual as mesmas estão localizadas associando-se o prefixo referente ao tipo de parâmetros que os dados armazenados se referem, conforme Tabela 1.

TABELA 1 – Prefixos identificadores dos tipos de recursos gerenciáveis.

Tipo de Recurso	*Prefixo*
Delegate Computer	*DC*
Inspector Computer	*IC*
Controlador	CTRL
Base de dados	*DBRO, DBNR, DBDR, DBHO, DBPO ou DBMR*
Agente *proxy*	APX
Agente *adapter*	AAD
MIB	*MIB*

Quanto ao controlador, há a possibilidade de existir mais de um sendo gerenciado por um mesmo *IC*, tal que sua nomenclatura deve herdar o nome do *IC* ao qual está conectado mais um prefixo determinado e correspondente sufixo. Quanto ao nome da *MIB*, deve-se identificar a qual controlador pertencem os objetos nela contidos, sendo indicado o nome do controlador com específico prefixo; os agentes-*proxy* e agentes-*adapter* devem indicar para qual controlador estão sendo convertidas as mensagens ou sobre qual *MIB* está atuando, também com respectivo prefixo, conforme Tabela 1.

Dada a coexistência de diversos recursos gerenciáveis em um único *IC*, a sugestão de organização se estende à localização dos mesmos. Cada conjunto individual de recursos pertinentes a um controlador deve estar em um repositório, cujo seu nome é semelhante ao do próprio controlador, conforme Figura 3.9, e os recursos compartilhados, como as bases de dados, devem ficar em um repositório identificado pelo nome do *IC*. Toda a hierarquia de repositórios deve estar definidas dentro de um repositório principal, denominado CONTRAM, conforme figura 3.10.

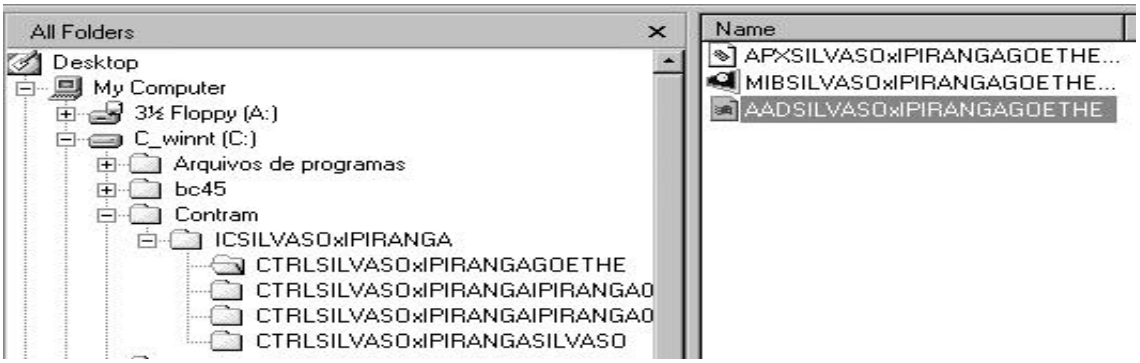

FIGURA 3.9 – Hierarquia de repositórios para um controlador e seus recursos.

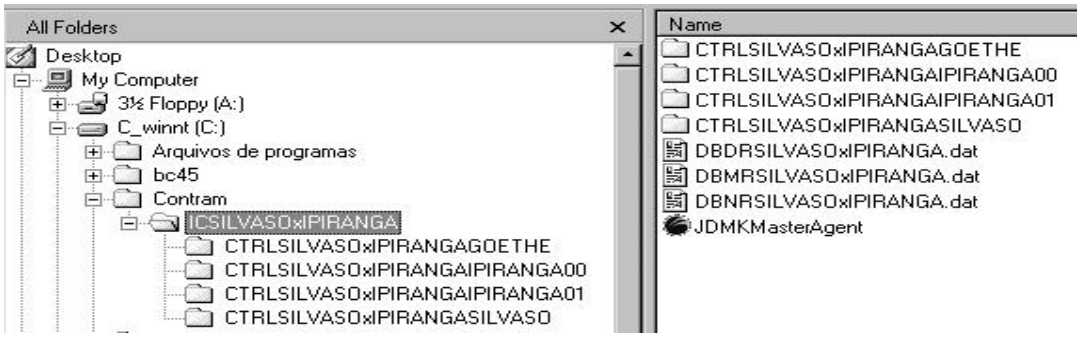

FIGURA 3.10 – Hierarquia de repositórios para um *IC* e seus controladores.

4. CONCLUSÕES

O CONTRAM atinge o seu objetivo de integrar diferentes controladores a um único SGTU. A seguir uma análise mais detalhada nos quesitos modularidade, longevidade e facilidade de integração.

Modularidade: com relação aos controladores, os fabricantes devem fornecer o agente-*proxy* correspondente. Uma vez esse agente-*proxy* ativado e o controlador já mapeado pelo *CONTRAM*, o mesmo está apto a receber dados. Caso haja troca de modelo de controlador, basta atualizar o agente-*proxy*. O *CONTRAM* interage com controlador referencial e este com os demais controladores da sua rede permitindo a integração entre redes de controladores diferentes. A limitação do *CONTRAM* está no fato de não interagir individualmente com os controladores. Para tanto seria necessário modificações na estrutura do *JDMK Agent*, conforme Figura 3.4, que deveria agregar mecanismos de agente gerente e não somente agente. Características da ferramenta *JDMK* permitem a criação de agentes com esse perfil com relativa facilidade, porém essa modificação no *JDMK Agent* agrega custos à solução.

Longevidade: uma grande e importante vantagem do *CONTRAM* é que toda a padronização proposta está explícita ao mesmo. Está armazenada em forma de *MIB* permitindo que características das inovações tecnológicas nos dispositivos de controle possam ser inseridas na mesma, sem que seja necessário modificações no *core* do CONTRAM. Qualquer dispositivo de controle que definir uma *MIB*, e fornecer um agente-*proxy* está apto a ser integrado ao CONTRAM, independentemente do seu avanço tecnológico. A sua limitação atual é integrar equipamentos de *hardware* que possuam dados complexos, como imagem ou grandes tabelas por exemplo, que não possam ser mapeados para objetos *MIB.*

Facilidade de integração: na interface com os controladores o *CONTRAM* utilizou uma tecnologia padronizada, estável, consolidada e largamente utilizas na indústria da Informática, minimizando as dependências tecnológicas, que foi o protocolo *SNMP* e objetos dispostos em formato *MIB*. Associado ao modelo de desenvolvimento de agentes, conforme Figura 3.4 e localização do agente, conforme Figura 3.8, não necessita-se de nenhuma modificação na arquitetura do *hardware* dos controladores para que o mesmo seja integrado. Os fabricantes devem utilizar um *software* compilador de *MIB* para gerar automaticamente como saída um agente a partir de uma entrada comum que é a *MIB* padronizada, tal que em um primeiro instante todos os agentes gerados serão iguais. O que difere um dos demais é justamente as diferentes características de cada controlador que serão customizadas através de linhas de código inseridas neste agente genérico que foi gerado. Por exemplo, uma empresa que possui cinco diferentes modelos de controladores, deverá possuir cinco diferentes agentes-*proxy*, um para cada modêlo, mas que foram gerados a partir de um agente genérico. A única exigência que se faz é que os agentes devam estar escritos na linguagem “C” ou Java.

REFERÊNCIAS BIBLIOGRÁFICAS

[1] Dailey, H.P. and Meyers, D. **A Structured Approach to Developing Distributed Network Applications for ITS Deployment**, Departament of Electrical Engineering, University of Washigton and Federal Highway Administration, Seattle, WA, 1997.

[2] David P. and McGinnis E. **Understanding SNMP MIBs**, Prentice Hall Inc, 1997, New Jersey.

[3] Digicon, I.C. **DIGICOM CDC 100 Manual de Operação**. Porto Alegre, dez. 1993. Relatório técnico.

[4] **System Architecture for ITS in Japan, Draft por Japan Government bodies**, Novembro/99, disponível por WWW em http://www.itsa.org/committe.nsf, em 02/2000.

[5] Rossetti, R.J.F. **Um Ambiente para Suporte à Simulação de Sistemas de Tráfego Urbano**, dissertação de mestrado, CPGCC, UFRGS, 1998.

[6] Stallings, W. **SNMP, SNMPv2, and RMON – Pratical Network Management**, Second Edition, Addison-Wesley Longman Inc., 1996.

[7] SUN MICROSYSTEMS, Inc. **Java Dynamic Management Kit White Paper**. Palo Alto, CA, 2000.

[8] Tesc, I.C. **Flexcon III, Manual de Operação**, julho de 1994.