SICSDA - Uma Arquitetura de *Software* Distribuída, Configurável e Adaptável Aplicada às Várias Missões de Controle de Satélites

Adriana Cursino Thomé
Instituto Nacional de Pesquisas Espaciais
drithome@bol.com.br

Mauricio G. V. Ferreira
Instituto Nacional de Pesquisas Espaciais
mauricio@ccs.inpe.br

João Bosco S. Cunha
Universidade Federal de Itajubá
bosco@unifei.edu.br

Resumo

Propõe-se neste trabalho uma arquitetura distribuída, configurável e adaptável para o software de controle de satélites (SICSDA). O objetivo de se propor esta arquitetura é permitir que o controle dos vários satélites possa ser feito usando-se o mesmo conjunto de máquinas, possibilitando que se possa escolher qual dos satélites deseja-se monitorar em um determinado instante. Outro fator importante é a necessidade de se ter uma arquitetura que permita que uma nova missão possa ser acomodada sem a necessidade de se criar um sistema específico para o satélite a ser lançado, fazendo com que o esforço necessário para adaptar o sistema a esse novo requisito seja minimizado.

Palavras-chave: modelos de objetos adaptáveis, metamodelos, sistemas distribuídos.

1. Introdução

O Instituto Nacional de Pesquisas Espacias (INPE), como uma das principais organizações envolvidas na evolução tecnológica espacial brasileira, assumiu a responsabilidade do lançamento e controle dos primeiros satélites brasileiros. O programa espacial brasileiro compreende o lançamento de quatro satélites, sendo os dois primeiros utilizados para coleta de dados (SCD1 (Sistema de Coleta de Dados 1) e SCD2 (Sistema de Coleta de Dados 2)), e os outros, para sensoriamento remoto (CBERS1 *(China Brazil Earth Research Satellite 1)* e CBERS2 *(China Brazil Earth Research Satellite 2)*), todos já lançados.

Para gerir todas as peculiaridades inerentes ao controle de um satélite, o INPE criou uma infra-estrutura robusta, composta pelo Centro de Controle de Satélites (CCS), por duas estações de rastreamento (Estações de Cuiabá e Alcântara), por uma rede de comunicação (RECDAS) e por aplicativos de software para o controle de satélites (SICS). As estações remotas localizadas em Cuiabá e Alcântara oferecem, juntamente com o CCS, o suporte em terra para o contro1e dos satélites em órbita [4].

Para cada satélite já lançado foi desenvolvido um aplicativo de software que permite o seu monitoramento em terra. Isso é necessário, pois cada satélite tem características próprias, que normalmente variam, mesmo que sutilmente, de um satélite para outro. O acesso a esses aplicativos está restrito aos controladores de satélites fisicamente localizados no CCS no INPE em São José dos Campos. Cada satélite lançado necessita, portanto, que seja destinada a ele uma máquina ou um conjunto de máquinas específicas onde um aplicativo específico para aquele determinado satélite é executado, auxiliando no recebimento de seus dados e monitoramento de seu estado interno.

Isto significa que para cada novo satélite a ser lançado, um aplicativo deverá ser desenvolvido ou adaptado para aquele satélite em especial, e máquinas deverão ser destinadas à execução desse *software* específico, implicando em um custo de desenvolvimento adicional a cada novo lançamento, tanto em termos de *hardware*, quanto em termos de *software*.

Este contexto nos leva a pensar na criação de um único sistema de *software* para o controle de satélites, que permita que os diferentes tipos de satélites possam ser monitorados de uma mesma máquina ou de um mesmo conjunto de máquinas. Ainda assim, a necessidade de uma adaptação no sistema, Poe exemplo, para incorporar características de um novo satélite a ser monitorado, traria uma série de dificuldades para a adaptação do software, ocasionando um grande esforço despendido para que as novas características pudessem ser incorporadas ao sistema de forma que a qualidade do mesmo fosse mantida.

Todas estas questões aliadas ao desejo crescente de se obter aplicações que evoluam à medida que o domínio evolui, fez com que se pensasse em construir aplicações mais configuráveis, flexíveis e adaptáveis, permitindo que o sistema pudesse se adaptar, mais facilmente, às novas necessidades do domínio, acompanhando a evolução dos requisitos, porém, mantendo sua qualidade.

Uma forma de se conseguir isto é mover certos aspectos do sistema, como regras de negócio, por exemplo, para o banco de dados, fazendo com que dessa forma, elas possam ser facilmente modificadas. O modelo resultante permite que o sistema possa se adaptar rapidamente às novas necessidades do domínio através de modificações nos valores armazenados no banco de dados, ao invés de modificações no código.

Isto encoraja o desenvolvimento de ferramentas que permitam que os especialistas do domínio introduzam novos elementos ao *software* sem a necessidade de programação adicional, e que façam mudanças em seus modelos de domínio em tempo de execução, reduzindo significantemente o tempo para incorporação de novos requisitos ao *software*.

Arquiteturas que podem dinamicamente se adaptar em tempo de execução a novos requisitos de usuários são chamadas de "arquiteturas reflexivas" ou "meta-arquiteturas. Uma "arquitetura de modelos de objetos adaptáveis *(Adaptive Object_Model Architecture)*" é um tipo particular de arquitetura reflexiva que abrange sistemas orientados a objeto que gerenciam elementos de algum tipo, e que podem ser estendidos para adicionar novos elementos [12].

Dessa forma, um modelo de objetos adaptável é um sistema que representa classes, atributos e relacionamentos como sendo metadados. Os usuários modificam os metadados (modelo de objetos) para refletir as mudanças no domínio. Essas mudanças modificam o comportamento do sistema. Em outras palavras, o sistema armazena o modelo de objetos em um banco de dados e o interpreta. Conseqüentemente, o modelo de objetos é ativo, e quando ele é modificado, o sistema muda imediatamente.

Assim sendo, os metadados são usados em modelos de objetos adaptáveis para descrever o modelo em si. Desde que o sistema consiga interpretar os metadados para construir e manipular as descrições das classes do modelo em tempo de execução, torna-se fácil adicionar novas classes ao modelo de objetos adaptável, e torná-las imediatamente disponíveis para os usuários [5].

Usar a abordagem dos modelos de objetos adaptáveis (AOMs) no desenvolvimento de sistemas pode amenizar alguns dos problemas que vêm sendo encontrados pelos desenvolvedores de *software*, principalmente em relação à flexibilidade, evolução e manutenção do sistema, permitindo que o custo total do desenvolvimento e manutenção possa ser reduzido drasticamente [6].

2. A arquitetura SICSDA

O objetivo da arquitetura SICSDA é permitir que o controle dos vários satélites possa ser feito usando-se o mesmo conjunto de máquinas, possibilitando que se possa escolher qual dos satélites deseja-se monitorar em um determinado instante. É importante ressaltar que não é possível, pelo menos por enquanto, monitorar mais de um satélite por vez, já que apenas a estação de Cuiabá encontra-se em operação.

Outro fator importante é a necessidade de se ter uma arquitetura que permita que uma nova missão possa ser acomodada sem a necessidade de se criar um sistema específico para o satélite a ser lançado, fazendo com que o esforço necessário para adaptar o sistema a esse novo requisito seja minimizado.

A arquitetura SICSDA modela a aplicação para o controle de satélites baseando-se nos modelos de objetos adaptáveis (AOMs). Isso significa que nesta arquitetura os objetos do domínio do problema, por exemplo; telemetria, telecomando, *ranging*; ao invés de estarem localizados no código que implementa a aplicação, estão armazenados em um banco de dados para que possam ser instanciados em tempo de execução. Isto significa que o sistema tem um código genérico implementado em uma linguagem de programação orientada a objetos, que representa o metamodelo para as classes do domínio do problema do Sistema de Controle de Satélites. Esse código genérico (metamodelo) deve ser capaz de acomodar os diferentes modelos de objetos (metadados) dos vários satélites.

A arquitetura SICSDA, principalmente por questões de tolerância às falhas, é distribuída, e as funcionalidades oferecidas pela aplicação; por exemplo, visualização de telemetria, emissão de telecomando podem estar distribuídas dentro de um domínio de rede pré-definido. Isso significa que os objetos da aplicação podem ser instanciados em máquinas diferentes na rede, ocasionando, portanto, uma distribuição do código do sistema. O *middleware* é responsável por prover a localização desses objetos, ou seja, em que máquina da rede eles estão disponíveis.

Pode-se dizer que a arquitetura SICSDA é adaptável porque é possível, em tempo de execução, alternar entre os metadados dos diversos satélites, ocasionando uma instancialização de um novo modelo de objetos em cima metamodelo cada vez que uma troca de contexto desse tipo for requisitada pelo usuário, ou seja, cada vez que se desejar controlar outro satélite.

Pode-se dizer ainda que a arquitetura SICSDA é adaptável, porque é capaz de acomodar possíveis mudanças no domínio do problema através da configuração apropriada dos metadados, permitindo que se possa acompanhar a evolução dos requisitos do domínio, e adaptando-se às necessidades dos usuários. Dessa forma, os especialistas do domínio (controladores de satélites e engenheiros de satélites) e os desenvolvedores do *software* podem adaptar o sistema para acomodar novas classes através da criação, em tempo de execução, dessas classes e seus atributos.

Pode-se dizer que a arquitetura SICSDA é configurável em relação às regras de negócio, pois é possível que novas regras de negócio (ou novos métodos) sejam associadas a uma classe do domínio em tempo de execução.

A Figura 1 ilustra a estrutura e o funcionamento da arquitetura SICSDA. A seguir são detalhados os elementos que aparecem na Figura 1.

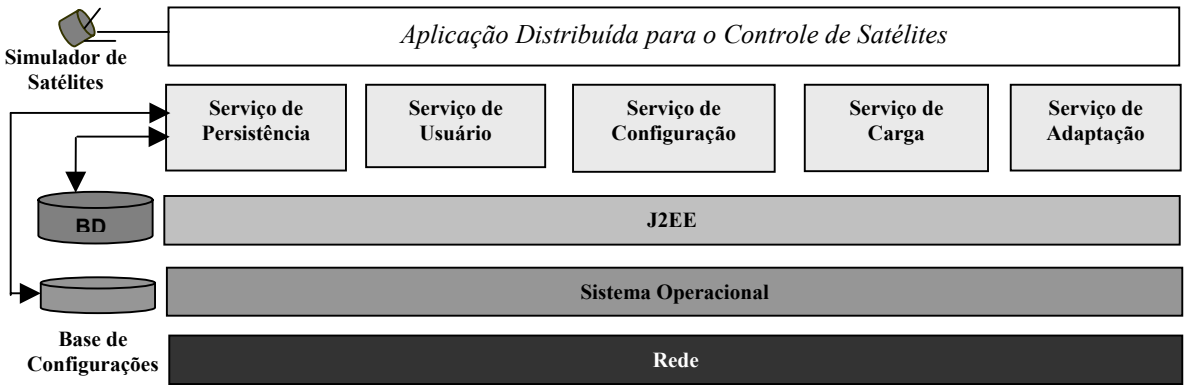

Fig.1 - Arquitetura SICSDA.

- **Aplicação Distribuída para o Controle de Satélites:** contém os objetos do *software* aplicativo que realiza o controle dos satélites (telecomando, *ranging*, telemetria etc.).
- **Serviço de Persistência:** é responsável por armazenar e recuperar do banco de dados os metadados do sistema. Adicionalmente, ele responsável por armazenar e recuperar de uma base de dados chamada de "Base de Configurações", os dados de autenticação dos usuários e os dados da configuração dos objetos nos nós onde a arquitetura SICSDA está implantada.
- **Serviço de Configuração:** faz parte da camada de apresentação do sistema e é responsável por manter os metadados do sistema e por manter a configuração dos objetos nos nós onde a arquitetura SICSDA está implantada. Deve oferecer aos especialistas do domínio e desenvolvedores de *software* uma interface adequada para que eles realizem tal tarefa.
- **Serviço de Carga:** é responsável por dar a carga inicial do sistema, ou seja, executar a rotina de carga do sistema nos nós onde a arquitetura SICSDA foi implantada.
- **Serviço de Usuário:** faz parte da camada de apresentação do sistema e é responsável por oferecer aos especialistas do domínio a interface adequada para a visualização de telemetria, emissão de telecomando, obtenção de medidas de distância e calibração etc, do satélite desejado. Além disso, ele deve prover aos usuários a interface para a autenticação no sistema.
- **Serviço de Adaptação:** é responsável por prover o metamodelo que permitirá com que efetivamente os objetos trazidos do banco de dados sejam instanciados, e que possam ser modificados em tempo de execução.
- **Simulador de Satélites:** é o *software* que simula a interação com os satélites, ou seja, que simula a chegada e envio de dados dos/para os satélites.

3. Resultados obtidos

Baseando-se no diagrama de classes que representa as classes do domínio do problema dos satélites SCD1, SCD2 e CBERS2, mostrado na Figura 2, pode-se obter um diagrama de classes genérico que representa o metamodelo para as classes do domínio do problema do Sistema de Controle de Satélites. Esse metamodelo é apresentado na Figura 3. Ele foi obtido através da aplicação de uma seqüência de *design patterns: pattern TypeObject, pattern Property, pattern Accountability* e *pattern Strategy.* Mais informações sobre esses *design patterns* podem ser encontradas em [5] e [12].

O ambiente para o protótipo da arquitetura SICSDA foi desenvolvido usando-se a linguagem *Java* versão 1.4.1, o banco de dados *Caché,* versão 5.0.5, o ambiente de desenvolvimento *Jbuilder* versão X, e o servidor de aplicações J2EE *Jboss* versão 3.0.8.

O metamodelo foi representado no banco de dados, e os metadados para cada satélite puderam ser armazenados através de uma interface gráfica disponibilizada pelo Serviço de Configuração. Através dessa interface, é possível realizar alterações em tempo de execução nas classes e seus atributos armazenados, e também associar novos métodos às classes.

O usuário interage com o sistema para a realização de alguma funcionalidade, por exemplo, Visualizar telemetria, Enviar Telecomando ou Obter Medidas, através da interface provida pelo Serviço de Usuário. Na Figura 4 pode-se visualizar o diagrama de seqüência que representa a realização dessas operações no sistema. Deve-se observar que, em virtude das classes do metamodelo serem genéricas, apenas um diagrama de seqüência foi necessário para representar a realização das operações citadas.

Apesar de muitos aspectos apontarem para o fato de que, pelos menos inicialmente, as arquiteturas baseadas em AOMs exigem mais esforço para serem construídas, pretende-se com este trabalho dar um passo significativo em direção a melhoria da reusabilidade e alterabilidade do Sistema de Controle de Satélites, já que os sistemas adaptáveis têm a característica de acompanhar mais facilmente a evolução dos requisitos do negócio.

Na verdade, o esforço para se realizar alterações em sistemas desse tipo pode ser bastante minimizado, já que alterações no código podem ser reduzidas substancialmente. Adicionalmente, com este tipo de arquitetura pode-se permitir que os próprios especialistas do domínio realizem algumas alterações, melhorando a alterabilidade e diminuindo a intervenção dos desenvolvedores do *software* na evolução do sistema.

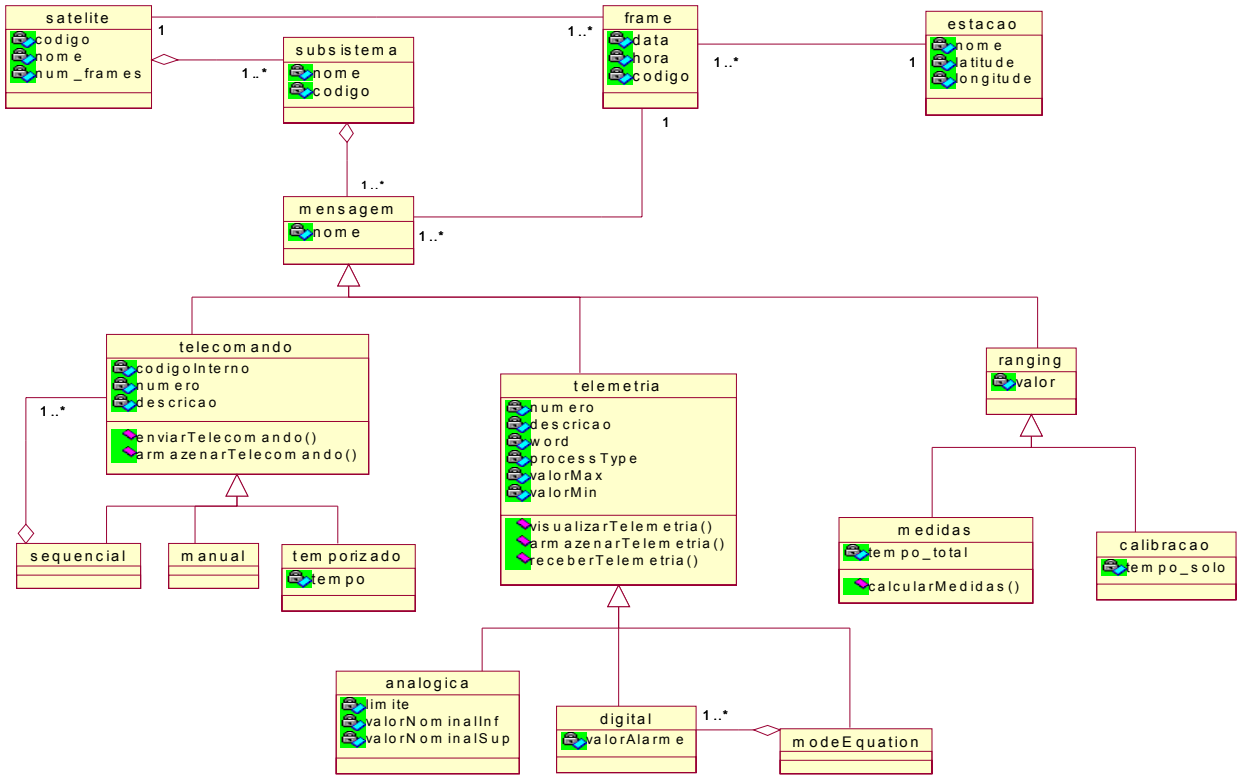

Fig. 2 – Diagrama de classes para os satélites SCD1, SCD2 e CBERS2.

Além disso, com esse tipo de arquitetura pode-se dar um passo em relação à melhoria do fator economicidade do Sistema de Controle de Satélites, uma vez que futuras missões poderão aproveitar quase todo o investimento de *hardware* e *software* já feito em outras missões.

O trabalho aqui descrito une, portanto, áreas que normalmente são exploradas de forma independente, trazendo para si uma característica multidisciplinar, ou seja, possibilitando a integração e colaboração da área de sistemas distribuídos, modelos de objetos adaptáveis e engenharia de *software*. Além disso, o trabalho aproveita esforços do passado através da elaboração de uma arquitetura que modela a aplicação para o controle de satélites com base nas arquiteturas SOFTBOARD e SICSD, propostas em [3] e [4], respectivamente.

Atualmente, o trabalho desenvolvido já foi apresentado e/ou publicado em [7], [8], [9], [10] e [11], e originou os trabalhos sendo desenvolvidos em [1] e [2].

Espera-se, desta forma, colaborar, mesmo que de forma modesta, para o avanço da pesquisa no Brasil, e para o sucesso da missão espacial brasileira, oferecendo uma nova alternativa para a arquitetura do *software* para o controle de satélites, e, principalmente, abrindo novos campos de estudo em direção aos sistemas adaptáveis.

4. Referências Bibliográficas

[1]Almeida, W. R. **Uma abordagem para a persistência dos modelos de objetos de sistemas distribuídos, configuráveis e adaptáveis.** Dissertação de Mestrado em Computação Aplicada. INPE. 2004. **(Em andamento)**

[2]Cardoso, P. E. **Modelo de objetos dinâmico aplicado ao processamento de telemetrias de satélites.** Dissertação de Mestrado em Computação Aplicada. INPE. 2004. **(Em andamento)**

[3]Cunha, J. B. S. **Uma abordagem de qualidade e produtividade para desenvolvimento de sistemas de software complexos utilizando a arquitetura de placa de software – SOFTBOARD.** Tese de Doutorado, Computação Aplicada. INPE. 1997.

[4]Ferreira, M. G. V. **Uma arquitetura flexível e dinâmica para objetos distribuídos aplicada ao software de controle de satélites.** Tese de Doutorado, Computação Aplicada. INPE. 2001.

[5]Johnson, R.; Yoder, J. W. The adaptive object-model architectural style. **IEEE/IFIP Conference on Software Architecture 2002 (WICSA3´02).** Canada. August of 2002.

[6]Ledeczi, A.; et al. Synthesis of self-adaptive software. **IEEE Areospace Conference Proceedings.** USA. V 4, pg. 501-507. 2000.

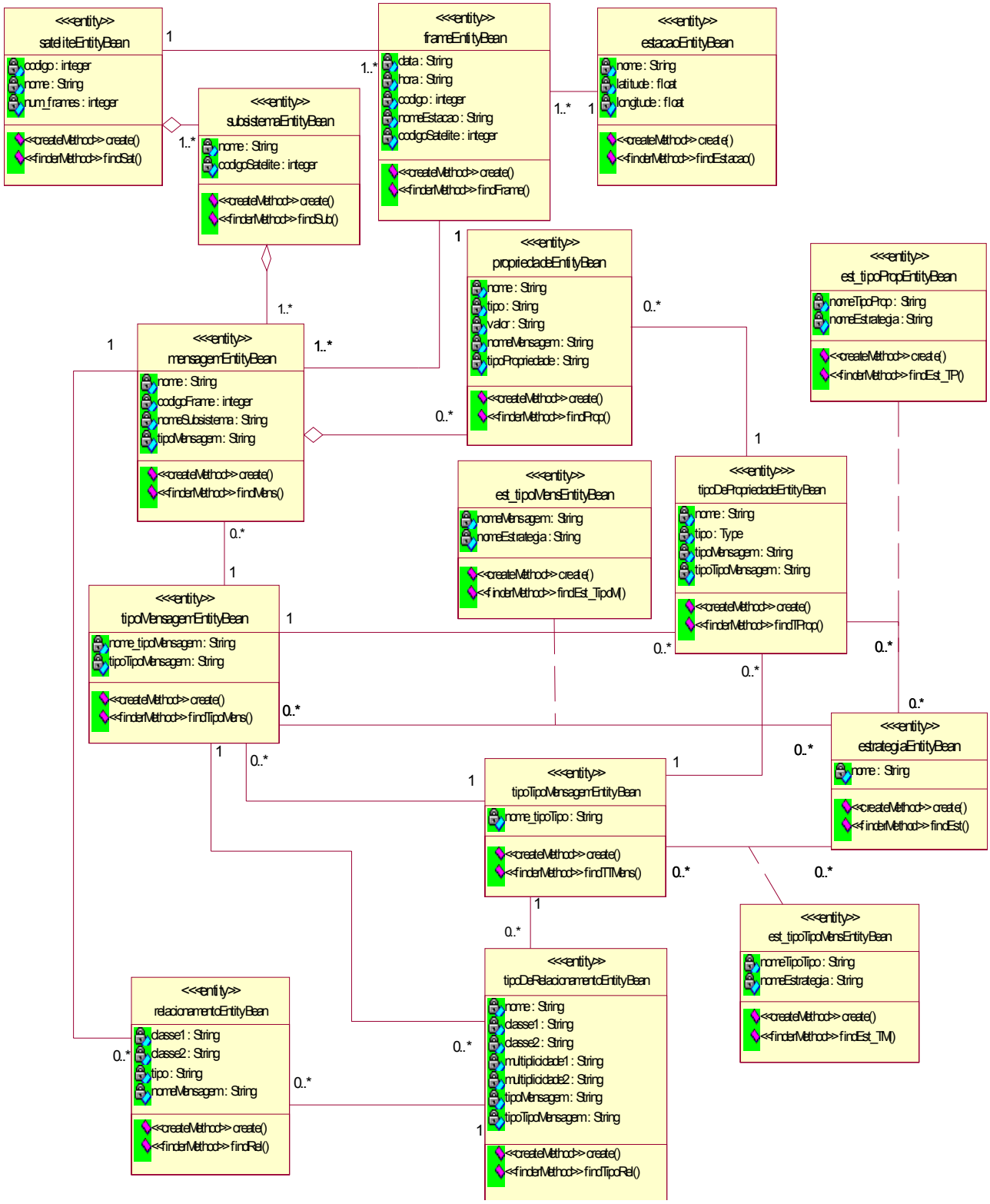

Fig. 3 – Diagrama de classes que representa o metamodelo para as classes do domínio do problema do sistema de controle de satélites.

[7]Thomé, A. C.; et al (a). SICSDA – An adaptive configurable distributed software architecture applied to satellite control missions. **XVII SBES - Simpósio Brasileiro de Engenharia de Software – VIII Workshop de Teses em Engenharia de Software.** Manaus, AM**.** October of 2003.

[8]Thomé, A. C.; et al (b). Uma arquitetura de software reflexiva baseada em modelos de objetos adaptáveis. **II Encontro de Informática de Campo Largo.** Campo Largo, PR. Novembro de 2003.

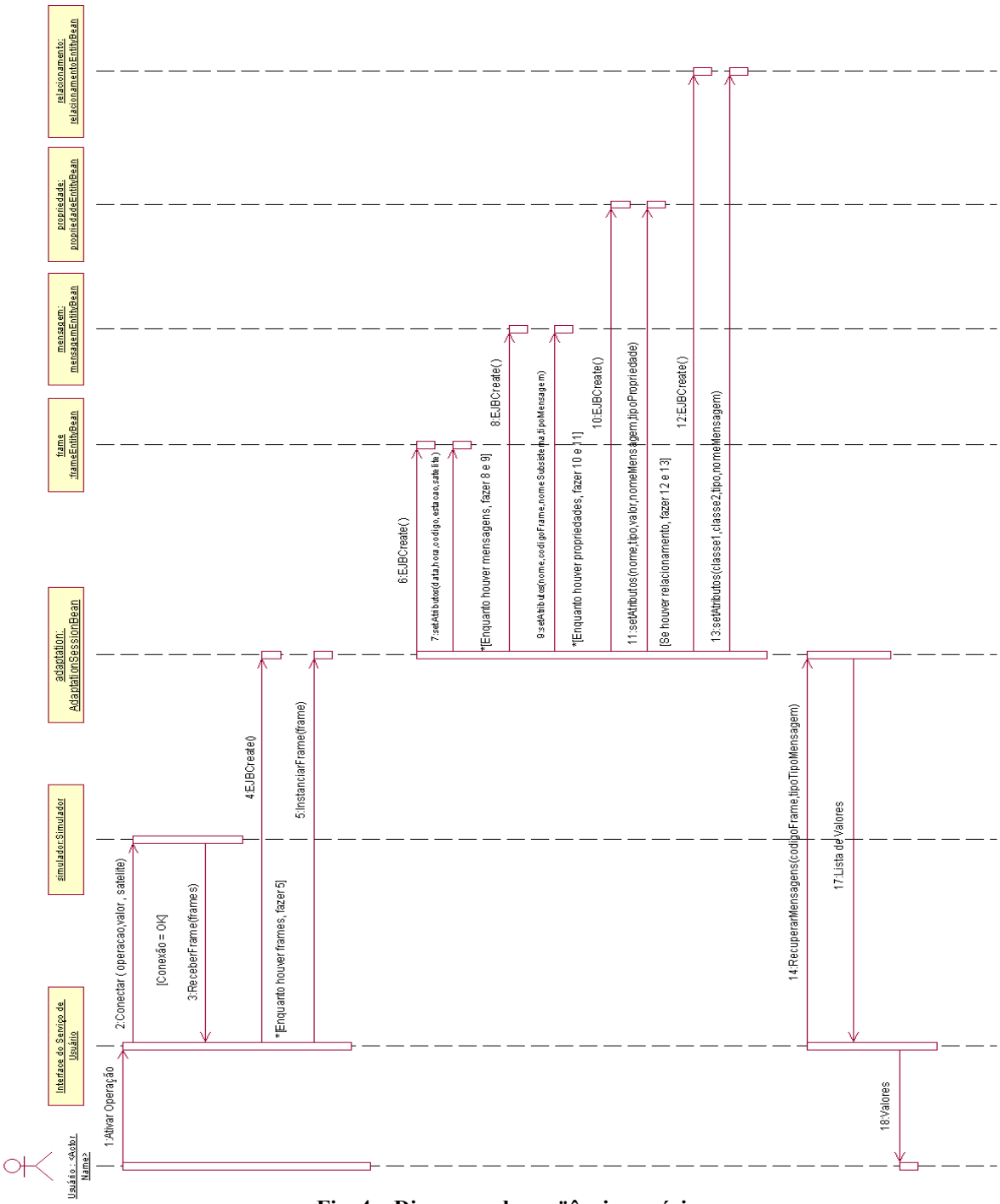

Fig. 4 – Diagrama de seqüência genérico.

[9]Thomé, A. C.; et al (c). Uma Arquitetura de Software Distribuída, Configurável e Adaptável Aplicada às Várias Missões de Controle de Satélites (SICSDA).**III Worshop dos Cursos de Computação Aplicada do INPE**. São José dos Campos, SP. Novembro de 2003.

[10]Thomé, A. C.; et al (d). Establishing an adaptive configurable distributed software architecture applied to satellite control missions. **Eighth International Conference on Space Operations – SpaceOps 2004**. Montreal, Canadá. May of 2004.

[11]Thomé, A. C.; et al (e). SICSDA: an Adaptive Configurable Distributed Software Architecture Applied to Satellite Control Missions. **18th European Conference on Object-Oriented Programming - ECOOP 2004.** Oslo, Norway. June of 2004.

[12]Yoder, J. W.; et al. Architecture and design of adaptive object-models. **ACM Sigplan Notices.** Vol. 36, Fasc. 12, pg. 50-60, December of 2001.