

LoCCAMConfigurator: Uma Ferramenta para Modelagem Visual de Configurações de um *Middleware* de Suporte à Aplicações Conscientes de Contexto

Paulo Artur de Sousa Duarte*, Francisco Anderson de Almada Gomes[†], Felipe Mota Barreto[†], Windson Viana*, Fernando Antonio Mota Trinta*

*Mestrado e Doutorado em Ciência da Computação (MDCC) - Departamento de Computação (DC)

[†]Departamento de Engenharia de Teleinformática

*Grupo de Redes de Computadores, Engenharia de Software e Sistemas (GREat) - Universidade Federal do Ceará (UFC) - Fortaleza – CE – Brasil

{pauloduarte, franciscoanderson, felipebarreto, windson, fernandotrinta}@great.ufc.br

ABSTRACT

The paper aims to describe the architecture and the main features of LoCCAMConfigurator, a tool for visual modeling of context information and contextual rules. This tool assists mobile application developers in the task of modelling context information from their applications and to define context-aware behaviors. LoCCAMConfigurator uses model-driven engineering approach for context modeling and subsequent automatic code generation. The tool uses the models created by the users in order to generate an Android project and a configured version of the context-aware middleware LoCCAM (Loosely Coupled Context Acquisition Middleware). This Android project includes the library and methods of communication between LoCCAM and the future application being developed. All the code for dealing with context gathering, filtering, detection and query is generated by the tool, then, developers can concentrate in the business part of their application.

General Terms

Performance, Human Factors, Languages.

Keywords

DSL, Middleware, MDE, Context-Aware

1. INTRODUÇÃO

Os dispositivos móveis, tais como *smartphones* e *tablets*, se tornaram dispositivos multimídia de utilidade diária e verdadeiros centros de informação sobre os usuários e seus hábitos. Sua evolução e popularização aumentaram a demanda por aplicações voltadas para estas plataformas. Por sua vez, estas plataformas permitiram uma nova gama de possibilidades para as aplicações, tornando-as capazes de, por exemplo, caracterizar o estado atual e as modificações do ambiente no qual o usuário está inserido (e.g., luminosidade) ou do próprio usuário (e.g. a localização atual deste). Isto se tornou conhecido como sensibilidade ao contexto ou ciência de contexto [1]. São denominadas aplicações móveis e sensíveis ao contexto aquelas que executam a maior parte de sua interação com o usuário em um dispositivo móvel e adaptam o seu

comportamento (e.g. conteúdo, interface do usuário) para o contexto atual do usuário e do ambiente.

A grande variedade de dispositivos, plataformas de desenvolvimento e interfaces de comunicação existentes acrescentam dificuldades na criação de aplicações móveis e sensíveis ao contexto. Entre estas dificuldades, podem ser citadas a complexidade no acesso de informações de sensores e as limitações inerentes à natureza dos dispositivos móveis. Exemplo de tais limitações seriam as restrições do consumo de recursos (e.g., a energia da bateria, a memória) e a ausência de perenidade na conexão do dispositivo com a Internet.

Objetivando-se a simplificação do desenvolvimento destas aplicações, surgiram propostas que utilizam plataformas de *middleware*, para realizar a intermediação entre a aquisição das informações contextuais, oriundas de sensores, *webservices* ou outras fontes, e as aplicações. Com isso, as plataformas de *middleware* uniformizam a sintaxe dos diversos dispositivos. Pode-se citar como exemplos de plataformas de *middleware* voltadas para este propósito, o *middleware* apresentado por Santos *et al* [4], o COPAL [5] e o LoCCAM (*Loosely Coupled Context Acquisition Middleware*) [1]. O LoCCAM é um *middleware* de aquisição de contexto voltado para dispositivos móveis com a plataforma Android. Sua arquitetura interna foi projetada para permitir um acoplamento fraco entre o *middleware* e as aplicações que o utilizam e a autoadaptação dos componentes que coletam e processam as informações contextuais.

Entretanto, o desenvolvimento de aplicações com o LoCCAM ainda traz alguns entraves. Dentre os quais, podem-se citar as dificuldades no tocante à configuração do *middleware* para um determinado dispositivo, com os componentes tendo que ser inseridos manualmente pelo desenvolvedor, e a sintaxe da comunicação entre a aplicação e o LoCCAM, que prescinde que o desenvolvedor conheça certos detalhes da arquitetura do *middleware*. Além disso, as sintaxes de certas ações utilizando o LoCCAM, tais como regras de eventos ou com composição de condições do tipo “uma OU outra”, permanecem complexas.

Este trabalho propõe-se a apresentar o LoCCAMConfigurator, uma ferramenta que utiliza os conceitos da MDE (*Model-Driven Engineering*) para simplificar o desenvolvimento de aplicações móveis e sensíveis ao contexto utilizando o LoCCAM e automatizar o processo de configuração do *middleware* nos dispositivos. O LoCCAMConfigurator é uma ferramenta para modelagem visual das informações e regras contextuais, que possibilita a geração do código envolvido na comunicação LoCCAM-aplicação e no acesso transparente às informações e regras contextuais especificadas. O

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

LoCCAMConfigurator utiliza como base o metamodelo DSL-LoCCAM, uma DSL voltada para o desenvolvimento de aplicações móveis e sensíveis ao contexto que utilizem plataformas de *middleware* para a aquisição contextual. O produto final do LoCCAMConfigurator é um projeto Android com os métodos de comunicação entre aplicação e LoCCAM gerados automaticamente, otimizando a transparência no uso das informações contextuais para o desenvolvedor.

O restante deste artigo está organizado do seguinte modo: na Seção 2, é introduzido o *middleware* LoCCAM, plataforma-alvo desta ferramenta. Na Seção 3, é apresentado o metamodelo da DSL-LoCCAM, através do qual são definidos os modelos na ferramenta. Na Seção 4, é descrito o *workflow* da LoCCAMConfigurator, detalhando o processo da modelagem à implementação da aplicação Android e, na Seção 5, são apresentadas suas principais funcionalidades. O artigo é finalizado com as considerações finais na Seção 6.

2. LoCCAM

O LoCCAM¹ (*Loosely Coupled Context Acquisition Middleware*) é uma infraestrutura de gerenciamento de contexto voltada para dar suporte a aplicações sensíveis ao contexto em dispositivos móveis (e.g., aplicações de anotações contextuais de fotos, aplicações de recomendação de conteúdo baseado na localização do usuário). O *middleware* intermedia de forma adaptativa a aquisição das informações contextuais e prover desacoplamento entre o código das aplicações e sensores de aquisição de contexto. O desacoplamento ocorre devido ao uso de CACs (Componentes de Aquisição de Contexto), que são os componentes responsáveis pela captura das informações contextuais. Cada CAC encapsula um sensor, que pode ser tanto um sensor físico (e.g., GPS), lógico (e.g., perfil do usuário) ou virtual (e.g., um serviço de meteorologia). O LoCCAM gerencia todos os CACs em execução, possibilitando a instalação, execução, parada e remoção destes através do uso do OSGi [1]. Como uma forma de possibilitar a comunicação entre o *middleware* e as aplicações, foi desenvolvido o conceito de *Context Keys*. É por meio do uso das *Context Keys* que as aplicações identificam as informações contextuais que desejam sem precisar especificar ou instanciar o código que provê a informação. Cada informação contextual é referenciada através de uma sequência de nomes separados por pontos (e.g., uma *Context Key* que sirva para a temperatura ambiente seria “context.ambient.temperature”). Outra parte importante do LoCCAM são os filtros baseados em espaço de tuplas. Esses filtros fornecem um mecanismo assíncrono de notificação e funcionam como regras de notificação (e.g, notifique a aplicação caso a temperatura esteja maior que 16°C) [2].

3. METAMODELO

O metamodelo utilizado como base para a modelagem das informações e regras contextuais é o DSL-LoCCAM, apresentado na Figura 1. O DSL-LoCCAM é baseado no metamodelo do MLContext [3], voltado para modelagem de contexto permitindo a reusabilidade dos modelos e independência de plataformas de *middleware*. No tocante aos elementos da definição das regras contextuais, o DSL-LoCCAM se baseou em conceitos apresentados pela abordagem de Santos *et al* [4], que é uma DSL voltada para o suporte de representação de regras e adaptação dinâmica do contexto. O metamodelo da DSL-LoCCAM tem

como base central a metaclassa Context, que representa e engloba todo o contexto existente em uma aplicação. Ele é formado por uma agregação de Regras (metaclassa Rules) e de Informações Contextuais (metaclassa ContextInformation). As Informações Contextuais são a parte principal da composição do Contexto, sendo a base que forma o Contexto e as Condições. As Regras são compostas por uma agregação de Condições (metaclassa Condition) e Ações (metaclassa Action). Quando uma regra tem mais de uma condição, as ações definidas são executadas se a composição das condições (“AND” ou “OR”) forem satisfeitas. Cada condição é formada por (i) uma informação contextual, (ii) o operador que indica o tipo de condição e (iii) o valor que quantifica a ação em si. Por sua vez, cada Ação é definida pelo tipo de ação que será executada por ela.

A DSL-LoCCAM, assim como todos os componentes da LoCCAMConfigurator, foi modelada com base em *frameworks* consolidados da IDE Eclipse. O metamodelo do DSL-LoCCAM foi construído com base no Ecore, através do EMF (*Eclipse Modeling Framework*)².

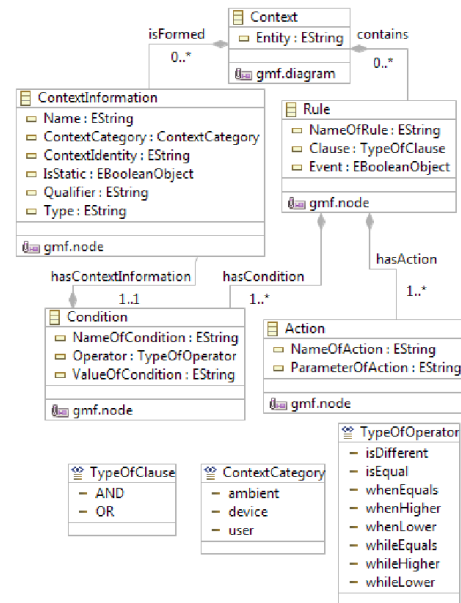


Figura 1 – Metamodelo do DSL-LoCCAM

4. WORKFLOW

O LoCCAMConfigurator pode ser dividido em duas partes principais: a modelagem visual do contexto e a geração de código com base no modelo criado. O componente que permite a modelagem visual da DSL-LoCCAM, apresentado na Figura 2, foi gerado utilizando o *framework* GMF (*Graphical Modeling Framework*)³, com o auxílio da ferramenta Eugenia⁴, ambos da IDE Eclipse. A geração de código é possibilitada graças ao uso *templates* de regras de transformação, que são transparentes ao desenvolvedor. Esses *templates* foram implementados utilizando xPand⁵, uma linguagem de transformação baseada em EMF, que

¹ <http://loccam.great.ufc.br/>

² <http://www.eclipse.org/modeling/emf/>

³ <http://www.eclipse.org/modeling/gmf/>

⁴ <http://www.eclipse.org/epsilon/doc/eugenia/>

⁵ <http://www.eclipse.org/modeling/m2t/?project=xpand>

permite a criação destes *templates* e a posterior execução da transformação que gerará o código. Com o objetivo de simplificar a sintaxe do código gerado, os *templates* foram implementados com base no uso da biblioteca LoCCAM_Lib. Esta biblioteca foi desenvolvida com o objetivo de simplificar a comunicação entre as aplicações e o *middleware*, encapsulando os métodos de comunicação com o LoCCAM e os CACs. Embora tenha logrado êxito parcial nesse objetivo, a LoCCAM_Lib ainda requer que o desenvolvedor conheça em detalhes a sintaxe do *middleware*, além de conhecer a *Context Key* de cada informação contextual que desejar implementar na aplicação. Uma visão geral do funcionamento do LoCCAMConfigurator, da modelagem até a aplicação sendo executada, é mostrado na Figura 3. A Figura ilustra um exemplo de uso da ferramenta na qual foi criada uma aplicação simples, que faz a leitura da temperatura a partir do sensor do dispositivo e apresenta na tela o valor desta em Celsius, Fahrenheit e Kelvin, caso todas estejam acima do valor “0” em suas respectivas escalas.

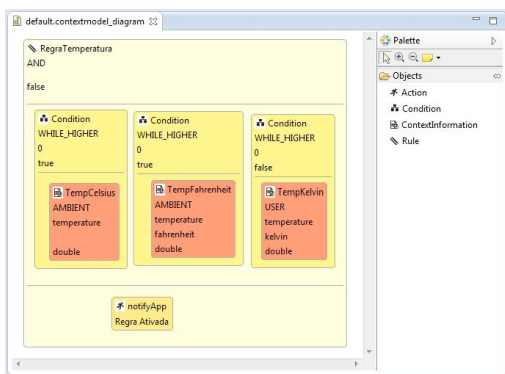


Figura 2 – Modelagem no LoCCAMConfigurator

Em (a), é realizada a modelagem visual da regra contextual desejada, especificando as condições e as informações contextuais esperadas. No caso, também são definidos os qualificadores da informação contextual. Os qualificadores definem atributos mais específicos da informação contextual, tais como unidades de medida, eixos de orientação ou quaisquer outros requisitos não funcionais. No caso da Figura3(a), ele especifica os valores da temperatura nas escalas Fahrenheit e Kelvin, uma vez que o padrão do sensor de temperatura ambiental do Android é a escala Celsius. Na Figura3(b), é mostrado os *templates* de regras de transformação que especificam como a modelagem realizada deve ser traduzida para a sintaxe que o LoCCAM possa utilizar. Estes *templates* são transparentes ao desenvolvedor, que não precisa criá-los. Com base no modelo apresentado por (a) e nesses *templates* existentes, é realizada a transformação e consequente geração do código. Na Figura3(a), é mostrado um exemplo do código gerado através da Figura3(a) e Figura3(b). O código gerado engloba duas classes Java. A primeira classe é a *LoccamActivator.java*, que instancia os métodos de comunicação com o LoCCAM, as regras e condições modeladas, além ter dos métodos para a comunicação e leitura de cada uma das informações contextuais desejadas. A segunda é uma *MainActivity.java*, que instancia o contexto da aplicação e inicializa a comunicação com o *middleware*. O produto final da transformação é um projeto Android, que tem ambas as classes geradas e a biblioteca LoCCAM_Lib. Com base neste projeto Android, a aplicação pode ser continuada, agora com o acesso simples e transparente às informações e regras contextuais já criadas, podendo ser utilizadas quando forem necessárias. Na

Figura3(d), é mostrada a aplicação Android desenvolvida com base na modelagem realizada em Figura3(a) e utilizando o código gerado em Figura3(c).

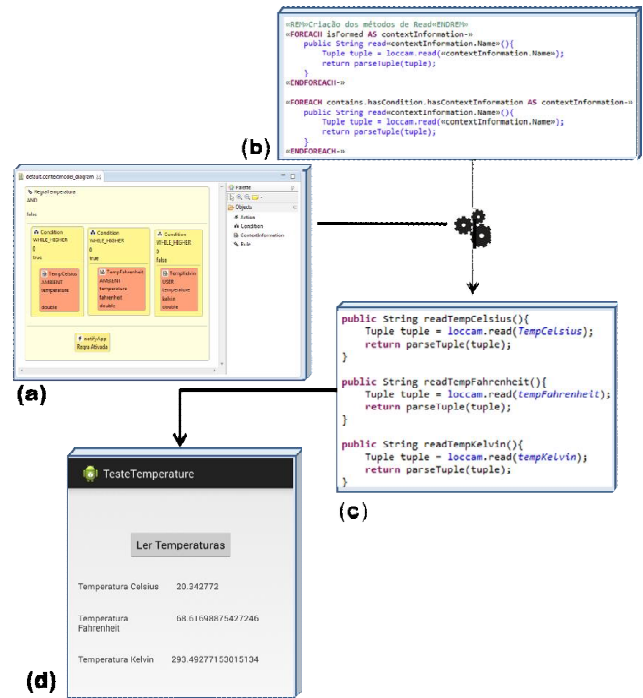




Figura 3 – Workflow do LoCCAMConfigurator

5. FUNCIONALIDADES

O LoCCAMConfigurator tem duas opções de transformação diferentes para um mesmo modelo: a primeira, exemplificada na Seção anterior e inicializada pelo botão , que visa a criação de uma aplicação Android que utilize o LoCCAM e com as regras contextuais já implementadas. A segunda opção, inicializada pelo botão , concerne à automatização da instalação do LoCCAM e instalação dos CACs correspondentes às informações contextuais especificadas no modelo. A Figura 4 ilustra o processo de configuração, com base no modelo mostrado na Figura 2. Uma vez que a modelagem tenha sido finalizada, como na Figura 4(a), e o modelo tenha sido salvo (sua extensão é “.contextmodel”), uma janela é aberta. Esta janela, mostrada na Figura 4(b), gerencia todo o processo de configuração. Esta janela pede para que o desenvolvedor selecione o modelo que ele deseja que seja utilizado para a configuração (e.g., o modelo recém salvo ou algum outro de extensão “.contextmodel”) e qual o dispositivo móvel ou emulador que deve ser o alvo da instalação. Uma vez que ambos estejam definidos, é realizada a instalação do LoCCAM no dispositivo-alvo e sua posterior inicialização. O passo seguinte é uma transformação com base no modelo da Figura 4(a) para a geração de um arquivo de texto simples com a lista das *Context Keys* equivalentes às informações contextuais modeladas pelo desenvolvedor. Esta lista é mostrada na Figura 4(c). Com base nesta lista e no dispositivo-alvo escolhido, é verificado se o dispositivo tem os sensores necessários para a execução das aplicações. Em caso positivo, a janela pede ao desenvolvedor que indique em qual diretório estão os CACs que ele deseja instalar.

A ferramenta vasculha esse diretório e realiza a triagem dos CACs que sejam correspondentes às *Context Keys* definidas na

lista. Quando encontradas, ela instala estes CACs nas pastas correspondentes. Caso algum CAC correspondente não seja encontrado no diretório especificado, a ferramenta realiza o *download* do CAC correspondente no repositório on-line de CACs. O processo de leitura, busca e instalação pode ser acompanhado via *console* pelo desenvolvedor, conforme mostrado na Figura 4(d). O tempo médio de execução de todo o processo é de 2117 ms, em um experimento realizado em um

Pentium Dual Core 2 GHz, com 2 GB de memória RAM DDR2 e com sistema operacional Windows 7 32bits. Na Figura 4(e), é mostrado o resultado final da instalação de configuração do LoCCAM, com os componentes especificados no modelos devidamente instalados de forma automatizada no dispositivo.

Um vídeo que exemplifica o uso do LoCCAMConfigurator pode ser acessado em:

<http://loccam.great.ufc.br/downloadFiles/video.mp4>

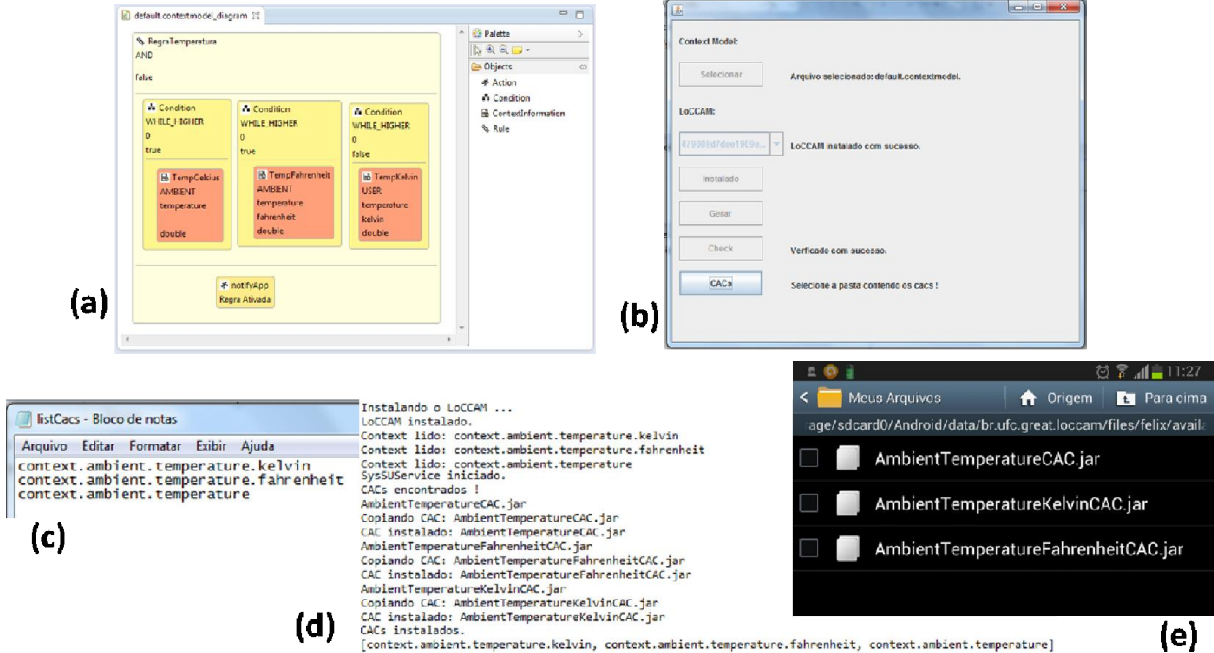


Figura 4 – Configuração automatizada do LoCCAM no dispositivo

6. CONCLUSÃO

Este artigo apresenta a LoCCAMConfigurator, uma ferramenta que permite a modelagem visual de informações e regras contextuais e, com base nestas, a geração automatizada do código de comunicação entre uma aplicação Android e o *middleware* LoCCAM. Ambos, LoCCAM e LoCCAMConfigurator, atuam em consonância com o intuito de diminuir a complexidade do código inerente ao acesso de sensores, sejam eles físicos, virtuais ou lógicos, e no aumento da transparência na aquisição e na utilização das informações contextuais pelo desenvolvedor.

Como trabalhos futuros, tem-se como objetivo um aprimoramento do LoCCAMConfigurator e do processo como um todo. Entre os aperfeiçoamentos planejados estão a inserção de ícones representando informações contextuais mais comuns (e.g., luminosidade) e condições comuns envolvendo informações contextuais de localização (distância de um certo ponto). Futuramente, objetiva-se que a criação de uma ferramenta que permita a modelagem visual completa da aplicação, com base no contexto modelado pela LoCCAMConfigurator, e com a posterior geração do código completo de uma aplicação Android.

7. AGRADECIMENTOS

Este projeto foi financiado parcialmente pela bolsa de mestrado da FUNCAP 2012-2014 e pelas bolsas PIBIC-UFC pelas bolsas dos projetos "Geração de aplicações móveis e sensíveis ao contexto utilizando Engenharia de Software baseada em Modelos" (2013-

2014) e "Geração de aplicações móveis e sensíveis ao contexto utilizando Engenharia de Software baseada em Modelos e o middleware LoCCAM" (2014-2015).

8. REFERÊNCIAS

- [1] M.E.F. Maia, A. Fonteles, B.J.A Neto, W. Viana, R.M.C. Andrade, "LoCCAM - Loosely Coupled Context Acquisition Middleware", In: 28th Symposium on Applied Computing (SAC), Coimbra, Portugal. March, 2013.
- [2] F.F.P. Lima, "SysSu - Um Sistema de Suporte para Computação Ubíqua". Dissertação de Mestrado, Departamento de Computação, Universidade Federal do Ceará, Fortaleza, CE, 2011.
- [3] J.R. Hoyos, J. García-Molina, J.A. Botía, "A domain-specific language for context modeling in context-aware systems". Journal of Systems and Software, Volume 86, Issue 11, November 2013
- [4] A.C. Santos, P.C. Diniz, J.M.P. Cardoso, D.R. Ferreira, "A Domain-Specific Language for the Specification of Adaptable Context Inference," International Conference on Embedded and Ubiquitous Computing (EUC), pp.268,273, 24-26 Oct. 2011
- [5] Fei Li, S. Sehic and S. Dustdar, "COPAL: An adaptive approach to context provisioning," IEEE 6th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob), pp.286,293, Oct. 2010