Suporte à Certificação de Componentes no Modelo de Representação X-ARM

Michael Schuenck[1], Glêdson Elias[1]

[1]Grupo COMPOSE (Component-Oriented Service Engineering)
Departamento de Informática - Universidade Federal da Paraíba (UFPB)
58.059-900 – João Pessoa – PB - Brasil
{michael, gledson}@compose.ufpb.br

Abstract. *Component repositories are commonly indicated as important resources for software reuse. In such repositories, in which independent producers and consumers can share software components, the evaluation of the quality of software components by third-party certifiers offers more security and confidence for consumers. In such a context, this paper introduces the resources supported by the XML-based Asset Representation Model (X-ARM) to describe component certifications. Such a model allows the description of certifications to be driven by distinct certification models, characterizing an important contribution since different certifiers can adopt distinct certification models.*

Resumo. *Repositórios de componentes são recursos comumente apontados como promotores do reuso de software. Nestes repositórios, onde produtores e consumidores independentes compartilham componentes de software, a avaliação da qualidade dos componentes por entidades certificadoras provê mais segurança e confiança aos consumidores. Neste sentido, este artigo introduz os mecanismos oferecidos pelo modelo de representação de componentes X-ARM (XML-based Asset Representation Model) para descrever certificações de componentes. Este modelo possibilita a descrição de certificações realizadas segundo diferentes modelos de certificação, caracterizando assim uma importante contribuição, pois diferentes entidades certificadoras podem adotar distintos processos de certificação.*

1. Introdução

Reuso de software é um importante mecanismo para incremento de produtividade e, consequentemente, redução de custos no desenvolvimento de software. Neste contexto, repositórios de componentes surgem com o potencial de facilitar ou mesmo habilitar o reuso de software ao possibilitar o compartilhamento, de forma livre ou não, de componentes de software entre diferentes produtores e consumidores [1]. Para que consumidores possam encontrar os componentes mais adequados às suas necessidades, repositórios realizam a indexação dos componentes, possibilitando que eles sejam posteriormente buscados por consumidores. Normalmente, a indexação se baseia em metadados dos componentes.

Por outro lado, conforme destacado por Bass [2], a inexistência de componentes de software com qualidade certificada constitui um inibidor do reuso de software, visto que consumidores temem que falhas desconhecidas sejam introduzidas em suas

aplicações. Diversas perspectivas, modelos e técnicas foram propostos e são utilizados para garantir a produção de software com qualidade. No entanto, no caso do reuso de componentes, existe a necessidade de um cuidado adicional com a qualidade, dado que este tipo de reuso normalmente se baseia na produção e no consumo de componentes por indivíduos que geralmente não se conhecem. Neste sentido, torna-se importante o papel de entidades certificadoras, sem relação com produtores de componentes, responsáveis por atividades de avaliação e certificação da qualidade dos componentes [3]. Um exemplo de entidade certificadora real é a *Underwriters Laboratories* [4].

Para a execução das atividades de certificação, um dos recursos mais conhecidos são modelos de certificação, que guiam as atividades de verificação dos componentes, provendo um mecanismo padronizado e repetível por meio da definição de um conjunto de características que devem ser avaliadas e pontuadas. O exemplo mais conhecido de modelo de certificação é o modelo ISO/IEC 9126-1 [5].

No entanto, apesar da existência de alguns modelos e técnicas de certificação de software [5][6][7][8][9], a maioria dos repositórios não provê mecanismos para garantir a qualidade de seus componentes e para permitir a busca com base em atributos de qualidade. Isto pode ser concluído com base no fato de que, dentre os modelos de representação de metadados de componentes encontrados na literatura [10][11][12], nenhum permite a representação de informações de certificação. Logo, a indexação e busca de componentes a partir de parâmetros qualitativos pode não ser possível. Por exemplo, um consumidor pode não ser capaz de encontrar componentes certificados de acordo com o modelo ISO/IEC 9126-1 ou componentes certificados e considerados de alta qualidade por mais de uma entidade certificadora.

Neste sentido, considerando a carência de iniciativas para descrição da certificação de componentes, este artigo introduz os recursos oferecidos pelo modelo X-ARM (*XML-based Asset Representation Model*) para representação de informações de certificação nos próprios metadados dos componentes. O modelo X-ARM é um modelo para representação de metadados de *assets* baseado no RAS [10]. No contexto do X-ARM, o termo *asset* designa um conjunto de artefatos criados ou reusados em um processo de desenvolvimento baseado em componentes. Cada conjunto de artefatos semanticamente relacionados é empacotado em um único arquivo, juntamente com um documento XML referente à descrição X-ARM, chamado manifesto.

O X-ARM foi projetado de forma a representar diversos tipos de informações importantes para o desenvolvimento de software baseado em componentes, sendo, portanto, um modelo bastante extenso. É importante ressaltar que os recursos providos pelo X-ARM para descrição de componentes já foram apresentados em [13]. No entanto, em [13], os recursos relacionados à certificação não foram discutidos, e, portanto, caracterizam o foco e a contribuição deste artigo.

Dentre os recursos do modelo X-ARM para a representação da certificação, se destacam como contribuição: a possibilidade de diferentes entidades certificadoras certificarem um mesmo componente; a viabilidade de uma entidade especificar e utilizar um modelo de certificação próprio; a habilidade de descrever certificações segundo quaisquer modelos de certificação; e a possibilidade de também descrever certificações dos próprios produtores dos componentes.

O restante deste artigo está dividido da seguinte forma: a Seção 2 apresenta as principais características de alguns modelos de certificação de software existentes; a

Seção 3 introduz as principais características do modelo X-ARM; a Seção 4 trata dos recursos deste modelo para representação de informações de certificação; a Seção 5 demonstra o uso do X-ARM na descrição de certificações; por fim, a Seção 6 apresenta algumas considerações finais.

2. Modelos de Certificação

A fim de possibilitar a especificação dos mecanismos oferecidos pelo modelo X-ARM para representação de informações de certificação, alguns modelos de certificação foram identificados, bem como as formas como são organizados. Assim, esta seção sumariza as conclusões desta revisão da literatura de forma a permitir, principalmente, a validação da abordagem de descrição de certificações oferecida pelo modelo X-ARM, conforme apresentado nas próximas seções.

Na literatura não existe uma grande variedade de modelos de certificação de software, sendo que o mais conhecido é o modelo ISO/IEC 9126-1 [5]. Este modelo se baseia na existência de características, sub-características e atributos. Os fatores de qualidade avaliados são classificados em seis características: *functionality*, *reliability*, *usability*, *efficiency*, *maintainability* e *portability*. Estas características são subdivididas em 27 sub-características. Por sua vez, as sub-características são decompostas em atributos, aos quais são associados os valores resultantes das certificações.

É importante destacar que o modelo ISO/IEC 9126-1 foi concebido para a certificação de software em geral. Dada sua aceitação, alguns esforços no sentido de adaptá-lo à certificação de componentes foram realizados. Exemplos destes esforços são os trabalhos de Bertoa e Vallecillo [6], de Álvaro et al. [7], e de Carvallo e Franch [8]. De forma simplificada, estes trabalhos alteram o modelo ISO/IEC 9126-1 tanto pela adição quanto pela remoção de características, sub-características e atributos, porém, mantendo a mesma organização hierárquica. Os dois primeiros trabalhos citados se concentram exclusivamente na definição de um modelo de certificação para componentes. Por outro lado, com uma motivação um pouco diferente, o trabalho de Carvallo e Franch estende o modelo ISO/IEC 9126-1 de forma a representar também fatores não técnicos de componentes, como informações de licença, custo e suporte, além de informações sobre o produtor do componente.

Como exemplo de uma abordagem não derivada do modelo ISO/IEC 9126-1, um *framework* para modelos de qualidade de componentes, denominado ABCDE, é apresentado em [9]. Este *framework* é definido por meio de cinco categorias de propriedades de avaliação: *acceptance*, *behavior*, *constraints*, *design* e *extension*. Por ser um *framework*, modelos de certificação podem ser criados a partir do ABCDE.

3. O Modelo X-ARM

O X-ARM é um modelo de metadados criado com o objetivo de propiciar o armazenamento, localização e recuperação não apenas de componentes, mas de *assets* em geral. Alguns exemplos de *assets* são casos de uso, diagramas UML, planos de teste, especificações de interfaces e de componentes, códigos-fonte e componentes.

Dentre as informações de um *asset* representadas pelo X-ARM, destacam-se o processo de desenvolvimento utilizado, certificados de qualidade, quem o certificou, o propósito do *asset*, qual modelo de componente é utilizado, quais interfaces são providas e requeridas, quais eventos são disparados e utilizados, quais componentes o compõe, e o relacionamento com outros *assets*. Apesar de ser bastante extenso, o

X-ARM define a representação de diversas informações como opcional, tornando-o facilmente adaptável a diferentes cenários e processos de desenvolvimento.

O X-ARM foi originalmente definido para o contexto de um repositório distribuído e compartilhado de componentes [14]. No entanto, ele também pode ser empregado por repositórios construídos com diferentes abordagens, já que a descrição de *assets* independe da arquitetura do repositório. Ao pressupor a existência de um repositório de componentes, a adoção do X-ARM apresenta alguns aspectos positivos, como, por exemplo, a validação da existência de *assets* semanticamente relacionados a um determinado *asset* e a possibilidade de oferecer um *front-end* para acompanhamento do processo de desenvolvimento de software baseado em componentes.

Por ser baseado no RAS, o X-ARM também utiliza o conceito de *profile*, cuja finalidade é separar conceitos, dividir complexidade e aumentar o reuso dos *assets*. O X-ARM define quatro *Profiles*: *Artifact Profile*, *Model Profile*, *Component Profile* e *Interaction Profile*. A Figura 1 ilustra a hierarquia de *profiles* do X-ARM.

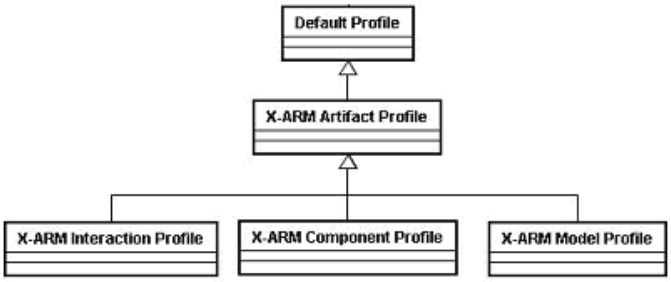

Figura 1: Hierarquia de Profiles do X-ARM

O *Artifact Profile* herda todos os elementos XML do *Default Profile*, definido pelo RAS, e acrescenta novos elementos, sendo o responsável por representar as funcionalidades básicas a todos os tipos de *assets* definidos pelo X-ARM.

O *Model Profile, Interaction Profile* e *Component Profile* estendem o *Artifact Profile*. O *Interaction Profile* descreve eventos, exceções e interfaces, que são os mecanismos usados por componentes para se comunicar com outros componentes ou aplicações. Com o *Component Profile* é possível descrever especificações e implementações de componentes, adicionando os elementos apropriados para descrever tais tipos de *assets*. Por fim, a intenção do *Model Profile* é padronizar os valores usados para representar características como certificação, classificação e negociação, habilitando buscas e comparações padronizadas entre diferentes *assets*.

4. Certificação de Componentes e Produtores no X-ARM

No modelo X-ARM, a certificação de um determinado *asset* é representada na forma de uma *descrição de certificação*, também denominada no modelo como certificado de qualidade. É importante ressaltar que um mesmo *asset* pode possuir diversas descrições de certificação, sendo cada uma delas produzida por diferentes entidades certificadoras.

Na hierarquia de *profiles* do X-ARM, as descrições de certificação de um determinado *asset* são representadas usando elementos do *X-ARM Artifact Profile*. É importante destacar que as descrições de certificação são representadas usando o *profile* mais básico, no caso o *X-ARM Artifact Profile*, ao invés de serem representadas com o *X-ARM Component Profile*. Como benefício desta abordagem, o X-ARM permite a

descrição da certificação de qualidade de qualquer tipo de *asset*, e não apenas de especificações e implementações de componentes de software, que são os tipos de *assets* que podem ser representados com o *X-ARM Component Profile*.

Para uniformizar ou padronizar as informações contidas nas descrições de certificação produzidas pelas diferentes entidades certificadoras, o X-ARM requer que tais entidades adotem *modelos de certificação*. No X-ARM, um modelo de certificação é representado usando elementos do *X-ARM Model Profile*. Assim, um modelo de certificação descrito com este *profile* também é um *asset*. A representação de modelos de certificação é de fundamental importância, pois permite a validação das informações contidas nas descrições de certificação dos *assets*, representadas com o *X-ARM Artifact Profile*. Desta forma, o X-ARM obriga que todas as descrições de certificação estejam em conformidade com seus respectivos modelos de certificação, que, por sua vez, são adotados pelas entidades certificadoras dos *assets*.

É importante ressaltar que, além de possuir elementos para representar a certificação de *assets*, o X-ARM também define elementos que permitem a representação da certificação dos produtores dos *assets*. Assim, de forma análoga ao adotado para *assets*, produtores possuem descrições de certificação que são produzidas por entidades certificadoras adotando modelos de certificação. Em resumo, no X-ARM, *assets* e produtores podem possuir diversas descrições de certificação que são geradas em conformidade com seus respectivos modelos de certificação.

Para não limitar os modelos de certificação que possam vir a ser adotados, e, assim, seja possível utilizar quaisquer processos de certificação, os modelos e as descrições de certificação de *assets* e produtores são representados na forma de pares chave/valor, sendo que as possíveis chaves e restrições para os valores são especificadas por *assets* de modelo de certificação.

As próximas seções apresentam como os modelos de certificação de *assets* e produtores podem ser representados usando o *X-ARM Model Profile*, e, em seguida, como as descrições de certificação de *assets* e produtores podem ser representadas usando o *X-ARM Artifact Profile*.

4.1. Representação de Modelos de Certificação

Em uma análise da estrutura dos principais modelos de certificação existentes, é fácil perceber que a maioria destes modelos adota estruturas hierárquicas para organizar as características e atributos avaliados durante o processo de certificação. Desta forma, no X-ARM, os modelos de certificação também são organizados em uma estrutura hierárquica de pares chave/valor. A Figura 2 ilustra os elementos do *X-ARM Model Profile* adotados para representar os modelos de certificação. Na Figura 2, a notação de classes da UML é usada para ilustrar os elementos e os atributos XML definidos pelo X-ARM para representar modelos de certificação.

O elemento <*asset*> é o elemento raiz das descrições X-ARM. Ele possui vários atributos e elementos filhos. Porém, dado o escopo deste trabalho, apenas os atributos e elementos relacionados à certificação de *assets* e suas funcionalidades são apresentados. Mais informações sobre os outros atributos e elementos podem ser encontradas em [13].

O atributo *id* do elemento <*asset*> é usado para identificar de forma única o modelo de certificação. Este atributo é importante para que outros *assets* possam indicar

o modelo de certificação que foi adotado em suas certificações. O nome do modelo de certificação é representado pelo atributo *name* do elemento <*asset*>.

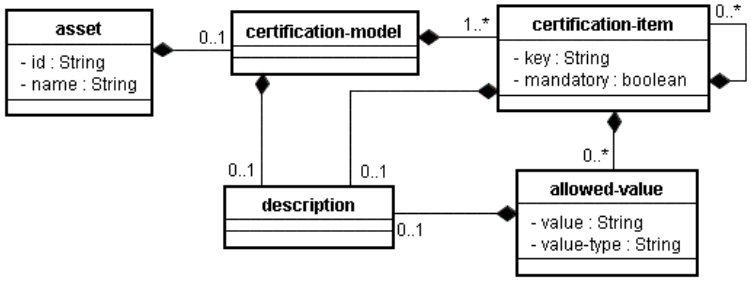

Figura 2: Especificação de modelos de certificação

Um modelo de certificação pode possuir um ou mais itens de certificação, que são os parâmetros verificados durante a avaliação da qualidade dos *assets*. Assim, o elemento <*certification-model*> possui um ou vários elementos <*certification-item*>. Em cada item, o atributo *key* identifica o nome de um parâmetro de certificação. Os itens de certificação que compõem um modelo podem ser obrigatórios ou opcionais. Para representar tal possibilidade, o atributo *mandatory* indica se o item especificado é obrigatório ou não em uma descrição da certificação de um *asset*. Além disso, um item pode se especializar em vários outros itens, criando assim uma estrutura hierárquica para o modelo de certificação.

Para cada item, opcionalmente, podem ser definidos os valores permitidos usando o elemento <*allowed-value*>, que possui os atributos *value* e *value-type*. O primeiro atributo indica um valor possível para o item de certificação e o segundo atributo indica o tipo deste valor, sendo possíveis apenas os tipos *string*, *integer, unsigned integer, float* e *unsigned float*. O atributo *value* pode ter um único valor discreto, múltiplos valores discretos, intervalos de valores numéricos ou uma combinação destas formas. Múltiplos valores discretos são representados pelos valores discretos separados por vírgula, por exemplo: "*yes, no*". Para representar um intervalo de valores numéricos, deve-se utilizar a seguinte notação: "*min_value..max_value*". Por exemplo, o valor "1..10" representa um intervalo de 1 a 10. Para especificar um intervalo de valor ilimitado utiliza-se o termo "*". Por exemplo, o valor "1..*" representa um intervalo de 1 até o limite superior permitido pelo respectivo tipo. No caso do tipo *string*, o termo "*" também indica que o item pode possuir qualquer valor. Por fim, é possível combinar valores discretos, múltiplos valores discretos e intervalos de valores, como por exemplo: "1..10, 25, 30..40, 50, 60".

Para facilitar o entendimento, todos os elementos de um modelo de certificação podem possuir uma descrição textual associada usando o elemento <*description*>.

É importante destacar que os modelos de certificação de *assets* e produtores são representados usando a mesma estrutura e elementos do *X-ARM Model Profile*. A Seção 5 apresenta exemplos de modelos de certificação de *assets* representados no X-ARM.

4.2. Representação de Descrições de Certificação

Depois que um modelo de certificação está descrito com a sintaxe definida pelo *X-ARM Model Profile*, é possível gerar descrições de certificação que adotam o modelo. Neste sentido, o *X-ARM Artifact Profile* define o elemento <*certification-info*>, cuja estrutura

é ilustrada na Figura 3. Este elemento é opcional, devendo ocorrer apenas em *assets* que possuem algum tipo de certificação.

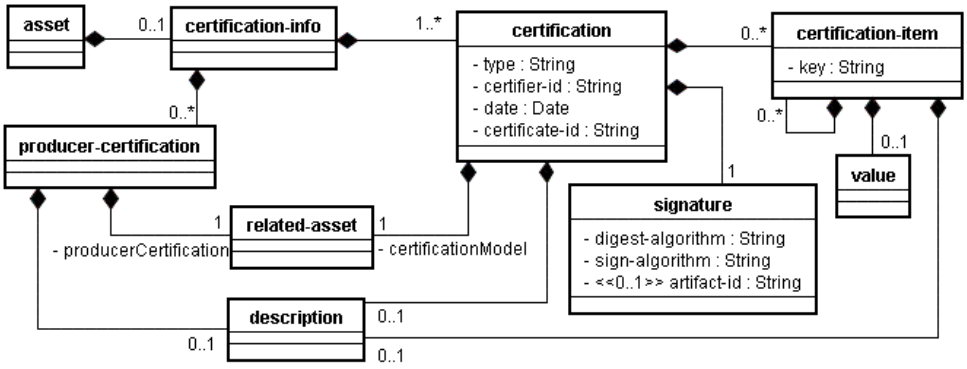

Figura 3: Especificação de descrições de certificação

Como pode ser observado na Figura 3, o elemento <*certification-info*> possui dois elementos: <*certification*> e <*producer-certification*>. O elemento <*certification*> é usado para representar informações da certificação do próprio *asset*. Por sua vez, o elemento <*producer-certification*> é usado para representar informações da certificação do produtor do *asset*.

Na descrição de certificação de um *asset*, como pode ser visto na Figura 3, o elemento <*certification*> inclui o tipo (*type*), a data da emissão da descrição de certificação (*date*), o identificador da entidade certificadora (*certifier-id*) e o identificador do próprio certificado (*certificate-id*). O tipo da certificação se refere ao nome do modelo de certificação adotado. O identificador da entidade certificadora corresponde ao esquema de nomeação usado pelo repositório de componentes para indicar a entidade que realizou a certificação do *asset*. O identificador do certificado é usado para identificar diferentes descrições de certificação de um determinado *asset*.

Cada elemento <*certification*> pode possuir elementos <*certification-item*>, responsáveis por descrever detalhes da certificação através de pares de chave/valor, representados pelo atributo *key* e pelo elemento <*value*>, respectivamente. É importante destacar que a estrutura de pares chave/valor deve seguir a hierarquia e as restrições impostas pelo respectivo modelo de certificação, que é identificado por meio de um elemento <*related-asset*>. Desta forma, um repositório de componentes pode validar as descrições de certificação, assegurando que as mesmas estão em conformidade com os respectivos modelos de certificação.

No sentido de assegurar que o certificado foi de fato emitido pela entidade certificadora especificada, o X-ARM define o elemento <*signature*>, que contém uma assinatura digital do certificado e as informações sobre os algoritmos utilizados na geração desta assinatura. Esta assinatura é gerada pela entidade certificadora considerando todos os arquivos incluídos no pacote do *asset* e o manifesto, ou seja, o documento XML com a descrição X-ARM do *asset*.

Na descrição de certificação de um produtor, como pode ser observado na Figura 3, o elemento <*producer-certification*> é usado para representar a certificação do processo de software adotado pelo produtor do *asset*. Esta representação é realizada apenas com uma referência a outro *asset*, criado especificamente para descrever a certificação do produtor. Esta referência é indicada usando o elemento <*related-asset*>.

Neste *asset*, a certificação do produtor é descrita da mesma forma que qualquer outro *asset*, usando o elemento <*certification*>. O benefício de manter estas descrições em *assets* separados é possibilitar o reuso das informações de certificação de um produtor nos diferentes *assets* que ele tenha criado.

De forma semelhante à representação de modelos de certificação, para auxiliar o entendimento, também podem ser associadas descrições textuais aos elementos <*certification*>, <*certification-item*> e <*producer-certification*> usando o elemento <*description*>.

4.3. Relacionamento entre Modelos e Descrições de Certificação

Uma vez que os elementos definidos pelo X-ARM para descrição de modelos de certificação e para descrição de certificações de *assets* e produtores foram apresentados de forma isolada, esta seção se dedica a explicar com mais detalhes o inter-relacionamento entre *assets* com diferentes papéis na representação de certificações. Assim, a Figura 4 ilustra um modelo conceitual onde este inter-relacionamento é identificado.

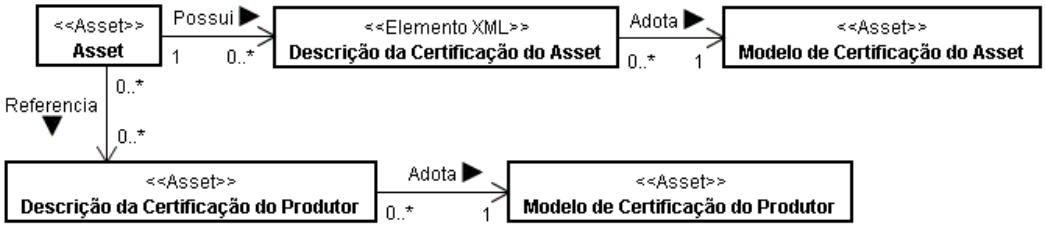

Figura 4: Modelo conceitual da representação de certificações

Com base na Figura 4, é possível perceber algumas características da representação de certificações. Primeiramente, um *asset* pode possuir diversas descrições de certificação, sendo cada uma emitida por diferentes entidades certificadoras e representada por um elemento <*certification*> em particular. Cada descrição de certificação se baseia apenas em um único modelo de certificação. Obviamente, para que as descrições de certificação de um *asset* sejam consistentes, elas não podem ter sido realizadas pela mesma entidade certificadora, exceto se estas certificações adotarem modelos de certificação distintos, cujas características e aspectos avaliados são diferentes. Por outro lado, um mesmo modelo de certificação pode ser adotado em diversas descrições de certificação de diferentes *assets*.

No caso de certificações de produtores, cada *asset* indica as diversas descrições de certificação do seu respectivo produtor. Assim como nas descrições de certificação dos *assets*, cada descrição de certificação do produtor deve ser gerada em conformidade com apenas um único modelo de certificação. De forma inversa, um modelo de certificação de produtor pode validar várias descrições de certificação de produtores.

4.4. Certificação em Repositórios X-ARM

Depois de apresentados os elementos para modelos e descrição de certificação, bem como o inter-relacionamento entre os *assets* envolvidos nas certificações, esta seção propõe um procedimento para certificação de *assets* e produtores. É importante mencionar que o procedimento é apresentado sob o ponto de vista de um repositório de componentes que adote o X-ARM como modelo de descrição dos seus *assets*.

Considerando que as certificações de *assets* e produtores são realizadas por entidades certificadoras, logicamente, as descrições de certificação são geradas pelas próprias entidades certificadoras. Por outro lado, o X-ARM não estabelece um ator específico com a responsabilidade de descrever os modelos de certificação. No entanto, no caso de entidades certificadoras que adotam processos de certificação proprietários, o mais natural é que as próprias entidades certificadoras descrevam os modelos de certificação adotados na certificação dos *assets*. Porém, é possível ainda que diferentes entidades certificadoras utilizem os mesmos modelos de certificação. Este é o caso de modelos de certificação públicos, definidos por instituições de padronização como a ISO. Assim, no caso de modelos de certificação públicos, a expectativa é que a própria instituição de padronização descreva o modelo de certificação. Após a instituição de padronização descrever um modelo de certificação público, qualquer entidade de certificação pode adotá-lo sem a necessidade de descrevê-lo novamente, bastando referenciá-lo com um identificador provido pelo repositório de componentes.

De forma resumida, o procedimento de certificação em um repositório que adote o X-ARM é constituído pela seqüência de atividades apresentada na Figura 5. Nesta figura, que representa um diagrama de atividades UML, considera-se que o *asset* a ser certificado é inicialmente descrito e registrado no repositório pelo seu produtor.

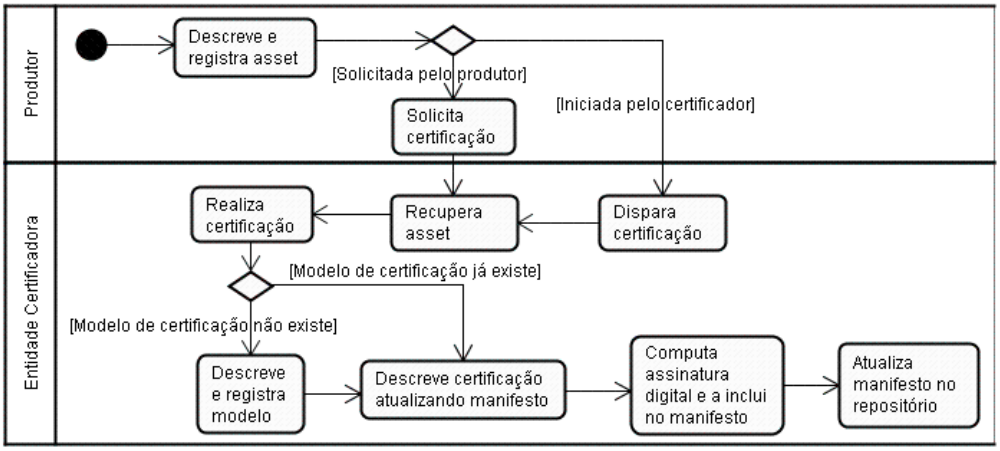

Figura 5: Diagrama de atividades envolvidas na certificação

Assim, a primeira atividade do diagrama ilustrado na Figura 5 representa a geração do manifesto de um *asset* e o subseqüente registro do *asset* no repositório. A partir de então, o processo de certificação do *asset* pode ser iniciado pela entidade certificadora. Em geral, a certificação pode ser disparada a pedido do próprio produtor ou por iniciativa unilateral da entidade certificadora. Na Figura 5, tais alternativas são representadas pelas atividades *solicita certificação* e *dispara certificação*.

Vale ressaltar que, na prática, existe um debate sobre a imparcialidade do processo de certificação adotar a abordagem requisitada pelo produtor ou disparada pela entidade certificadora. No entanto, o X-ARM não se limita a uma abordagem específica, deixando sob a responsabilidade de cada repositório selecionar a abordagem mais adequada às necessidades de seus respectivos produtores.

Uma vez que a entidade certificadora identifica a necessidade de certificar um dado *asset*, esta entidade recupera o *asset* do repositório, e, guiando-se pelo modelo de certificação adotado, a entidade certificadora realiza a certificação do *asset*.

Em seguida, a entidade certificadora deve gerar a descrição da certificação com o *X-ARM Artifact Profile*. Para gerar a descrição da certificação, existe a pré-condição do *asset* que representa o modelo de certificação adotado já ter sido previamente cadastrado no repositório. Assim, se o modelo de certificação ainda não existe, ele deve ser primeiramente descrito usando o *X-ARM Model Profile*, conforme apresentado na Seção 4.1.

Depois que a descrição da certificação do *asset* for concluída, a assinatura digital da certificação é computada, utilizando-se as definições apresentadas na Seção 4.2 para o elemento <*signature*>, e adicionada ao manifesto. Somente após a atualização do manifesto no repositório é que o *asset* pode ser considerado certificado.

5. Exemplos de Uso de Modelos e Descrições de Certificação

Nesta seção, exemplos de modelos e descrições de certificação de *assets* são apresentados. O objetivo é evidenciar o potencial do modelo X-ARM para representar diferentes modelos de certificação e instanciar tais modelos para gerar as respectivas descrições de certificação dos *assets*. Vale ressaltar que, por limitação de espaço, exemplos de modelos e descrições de certificação de produtores não serão apresentados. No entanto, tais modelos e descrições de certificação de produtores podem ser especificados de forma bastante semelhante.

A partir dos elementos definidos pelo *X-ARM Model Profile*, apresentados na Seção 4.1, a Figura 6 apresenta um exemplo de representação parcial do modelo de certificação ISO/IEC 9126-1 [5].

```
1. <asset id=" org.iso.isoiec91261-1.0" name="ISO/IEC 9126-1" ... >
…
2.     <certification-model>
3.         <certification-item key="Functionality" mandatory="false">
4.             <certification-item key="Accuracy" mandatory="false">
5.                 <certification-item key="Correctness" mandatory="false">
6.                     <allowed-value value="0..100" value-type="integer"/>
7.                 </certification-item>
8.             </certification-item>
9.             <certification-item key="Security" mandatory="false">
10.                <certification-item key="Data Encryption" mandatory="false">
11.                    <allowed-value value="yes, no" value-type="string"/>
12.                </certification-item>
13.                <certification-item key="Controllability" mandatory="false">
14.                    <allowed-value value="0..100" value-type="integer"/>
15.                </certification-item>
16.                <certification-item key="Auditability" mandatory="false">
17.                    <allowed-value value="yes" value-type="string"/>
18.                    <allowed-value value="no" value-type="string"/>
19.                </certification-item>
20.            </certification-item>
21.        </certification-item>
22.        <certification-item key="Efficiency" mandatory="false">
23.            <certification-item key="Resource Behavior" mandatory="false">
24.                <certification-item key="Memory Utilization" mandatory="false">
25.                    <allowed-value value="*" value-type="integer"/>
26.                </certification-item>
27.                <certification-item key="Disk Utilization" mandatory="false">
28.                    <allowed-value value="*" value-type=" integer "/>
29.                </certification-item>
30.            </certification-item>
31.        </certification-item>
…
32.    </certification-model>
33. </asset>
```

Figura 6: Modelo de Certificação ISO/IEC 9126-1

Na Figura 6 são apresentadas partes das características *Functionality* (linhas 3 a 21) e *Efficiency* (linhas 22 a 31). No primeiro caso, estão representadas as sub-características *Accuracy* (linhas 4 a 8) e *Security* (linhas 9 a 20). A primeira é subdividida em *Correctness* (linhas 5 a 7), que tem como valores permitidos números inteiros entre *0* e *100*. Por sua vez, a sub-característica *Security* tem os atributos *Data Encryption*, *Controllability* e *Auditability*. *Data Encryption* (linhas 10 a 12) e *Auditability* (linhas 16 a 19) definem os valores "*yes*" e "*no*" como permitidos. Já o item *Controllability* (linhas 13 a 15) tem números inteiros entre *0* e *100* como permitidos. Por fim, a característica *Efficiency* (linhas 22 a 31) apresenta o item *Resource Behavior* (linhas 23 a 30), subdividida nos atributos *Memory Utilization* (linhas 24 a 26) e *Disk Utilization* (linhas 27 a 29), que podem possuir valores inteiros.

Em três pontos do exemplo foram colocadas reticências para simplificar o exemplo. Na primeira e na última linha, as reticências indicam a existência de outros atributos e outros elementos que descrevem o elemento <*asset*>, respectivamente. Em ambos os casos, as informações omitidas estão fora do contexto deste artigo. Na terceira ocorrência de reticências, na linha 31, foram omitidos os demais itens de certificação definidos pelo modelo ISO/IEC 9126-1. Reticências também são utilizadas nos exemplos subseqüentes com finalidades similares.

Com base no exemplo da Figura 6, a Figura 7 apresenta um exemplo de descrição de certificação de um *asset* usando o modelo ISO/IEC 9126-1.

```
…  1. <certification-info>
2.          <certification type="ISO/IEC 9126-1" certifier-id="br.compose"
                        certificate-id=”br.compose.certificateA” date="2007-05-10">
3.              <related-asset id="org.iso.isoiec91261-1.0" name=”ISO/IEC 9126-1”
                        relationship-type="certificationModel"  required="true"/>
4.              <certification-item key="Functionality">
5.                  <certification-item key="Accuracy">
6.                      <certification-item key="Correctness">
7.                          <value>87</value>
8.                      </certification-item>
9.                  </certification-item>
10.                 <certification-item key="Security">
11.                     <certification-item key="Data Encryption">
12.                         <value>yes</value>
13.                     </certification-item>
14.                     <certification-item key="Controllability">
15.                         <value>98</value>
16.                     </certification-item>
17.                     <certification-item key="Auditability">
18.                         <value>yes</value>
19.                     </certification-item>
20.                 </certification-item>
21.             </certification-item>
22.             <certification-item key="Efficiency">
23.                 <certification-item key="Resource Behavior">
24.                     <certification-item key="Memory Utilization">
25.                         <value>3000000</value>
26.                     </certification-item>
27.                     <certification-item key="Disk Utilization">
28.                         <value>1200000</value>
29.                     </certification-item>
30.                 </certification-item>
31.             </certification-item> …
32.             <signature digest-algorithm="SHA1" sign-algorithm="RSA">
                    2fd4e1c67a2d28fced849ee1bb76e7391b93eb12
33.             </signature>
34.         </certification>
35. </certification-info> ...
```

Figura 7: Descrição de certificação baseada no ISO/IEC 9126-1

No exemplo da Figura 7, observa-se na linha 2 que o tipo de certificação adotado foi o *ISO/IEC 9126-1* e que a certificação foi realizada pela entidade certificadora cuja identificação é *br.compose*. Neste exemplo, como pode ser visto na linha 3, o modelo de certificação adotado está especificado no *asset* identificado por *org.iso.isoiec91261-1.0*. Seguindo as restrições deste modelo, foram incluídas algumas ocorrências do elemento <*certification-item*>, que descrevem a certificação do *asset*. Por exemplo, as sub-características *Correctness*, *Data Encryption*, *Controllability*, *Auditability*, *Memory Utilization* e *Disk Utilization* possuem os valores *87*, "*yes*", *98*, "*yes*", *3000000* e *1200000*, respectivamente. Por fim, o elemento <*signature*> (linhas 32 a 33) representa a assinatura digital emitida pela entidade certificadora, além dos algoritmos utilizados para gerar a assinatura, neste caso o SHA1 e o RSA. Neste exemplo, apenas o *asset* foi certificado, mas não o seu produtor, pois não existe nenhuma ocorrência do elemento <*producer-certification*>.

Para ilustrar a capacidade do modelo X-ARM descrever certificações realizadas a partir de diferentes modelos de certificação, a Figura 8 apresenta a especificação parcial de um outro modelo de certificação, sendo neste caso baseado no *framework* ABCDE [9]. Neste exemplo, nota-se que foram representadas as categorias *Acceptance* (linhas 3 a 7), *Constraints* (linhas 8 a 15) e *Extension* (linhas 16 a 20) e alguns de seus itens de avaliação. Os itens *Published evaluation* (linhas 4 a 6) e *Platform spec* (linhas 9 a 11) podem possuir qualquer valor textual. Já item *Ease of use* (linhas 12 a 14) pode possuir valores inteiros entre *0* e *100*, enquanto que o item *Portable across platforms* (linhas 17 a 19) pode possuir apenas os valores "*true*" e "*false*". Como pode ser observado no atributo *mandatory*, todos estes itens estão especificados como não obrigatórios.

```
1. <asset id="br.compose.abcde-1.0" name="ABCDE" ... > …
2.     <certification-model>
3.         <certification-item key="Acceptance" mandatory="false">
4.             <certification-item key="Published evaluation" mandatory="false">
5.                 <allowed-value value="*" value-type="string"/>
6.             </certification-item>
7.         </certification-item>
8.         <certification-item key="Constraints" mandatory="false">
9.             <certification-item key="Platform spec" mandatory="false">
10.                <allowed-value value="*" value-type="string"/>
11.            </certification-item>
12.            <certification-item key="Ease of use" mandatory="false">
13.                <allowed-value value="0..100" value-type="integer"/>
14.            </certification-item>
15.        </certification-item>
16.        <certification-item key="Extension" mandatory="false">
17.            <certification-item key="Portable across platforms" mandatory="false">
18.                <allowed-value value="true, false" value-type="string"/>
19.            </certification-item>
20.        </certification-item> …
21.    </certification-model>
22. </asset>
```

Figura 8: Modelo de certificação baseado no framework ABCDE

Por fim, a Figura 9 ilustra um exemplo de descrição de certificação baseada no modelo ABCDE, apresentado na Figura 8. No exemplo, observa-se na linha 2 que o tipo de certificação adotado foi o *ABCDE* e que a certificação foi realizada pela entidade certificadora cuja identificação é *br.compose*. Como pode ser visto na linha 3, o modelo de certificação adotado está especificado no *asset* indicado por *br.compose.abcde-1.0*. Dado que este modelo define os itens *Acceptance* e *Published evaluation* como não obrigatórios, estes itens não foram descritos. Todos os demais itens foram representados

seguindo as restrições deste modelo. Por exemplo, as sub-características *Platform spec*, *Ease of use* e *Portable across platforms* possuem os valores "*OS: Windows 98+, Processor: x86, 32 bits*", *80* e "*false*". Por fim, o elemento <*signature*> (linhas 17 a 18) representa a assinatura digital emitida pela entidade certificadora.

```
… 1. <certification-info>
2.    <certification type="ABCDE" certifier-id="br.compose"
                    certificate-id=”br.compose.certificateB” date="2007-05-12">
3.       <related-asset id=" br.compose.abcde-1.0" name=”ABCDE”
                    relationship-type="certificationModel"  required="true"/>
4.      <certification-item key="Constraints">
5.         <certification-item key="Platform spec">
6.              <value>OS: Windows 98+, Processor: x86, 32bits</value>
7.         </certification-item>
8.         <certification-item key="Ease of use">
9.              <value>80<value>
10.         </certification-item>
11.      </certification-item>
12.       <certification-item key="Extension">
13.          <certification-item key="Portable across platforms">
14.             <value>false</value>
15.          </certification-item>
16.      </certification-item>
17.      <signature digest-algorithm="SHA1" sign-algorithm="RSA">
              3bc5a3b46d3e27acdf961bc2be35a1842b94bd312
18.      </signature>
19.   </certification>
20. </certification-info> ...
```

Figura 9: Descrição de certificação baseada no framework ABCDE

6. Considerações Finais

O modelo X-ARM foi criado com o objetivo de descrever os principais tipos de informações referentes ao desenvolvimento de software baseado em componentes, incluindo, por exemplo: processo de desenvolvimento, classificação, formas de negociação, informações de qualidade e os próprios componentes, por meio de suas interfaces providas e requeridas, seus eventos disparados e recebidos. Trata-se, portanto, de um modelo rico e extenso. Logo, dada a limitação de espaço, este artigo se limitou a apresentar seus mecanismos para descrição de certificações de componentes.

Um importante aspecto positivo do X-ARM é não limitar-se à descrição de certificações de componentes de acordo com um único modelo de certificação específico. Para tal, o X-ARM define que não apenas descrições de certificação sejam representadas, mas também, seus respectivos modelos de certificação, que habilitam a validação das certificações. Como vantagem adicional desta abordagem, tem-se a padronização das informações de certificação, já que, apesar da flexibilidade, todas as informações seguem uma estrutura e sintaxe pré-definida. Por outro lado, um aspecto negativo do X-ARM é a necessidade do esforço adicional requerido para também representar os modelos de certificação. No entanto, este aspecto não chega a ter um impacto negativo tão forte, visto que, tendo sido representado, um dado modelo de certificação pode ser reusado e referenciado por inúmeras descrições de certificação.

As facilidades de representação de informações de certificação do X-ARM constituem uma importante contribuição para o contexto de repositórios de componentes. Esta contribuição torna-se ainda mais relevante quando considerado que os outros modelos de representação de componentes existentes, tais como OSD [11], CDML [12] e o próprio RAS [10], não tratam da questão de certificação. Portanto, ao contrário do X-ARM, estes outros modelos não permitem aos consumidores realizar

buscas ou selecionar *assets* com base em informações de certificação, reduzindo a segurança e confiança dos mesmos, e, assim, dificultando o reuso em larga escala.

Embora as contribuições do X-ARM sejam importantes, ainda existem algumas melhorias que podem ser incluídas. Por exemplo, na especificação de modelos de certificações, é possível enriquecer as regras para restrição dos valores permitidos adotando expressões regulares. Outros esforços futuros serão concentrados em ferramental para preenchimento e validação automática de certificações de componentes usando os respectivos modelos de certificação. Desta forma, para facilitar o preenchimento, tal ferramental pode adotar formulários dinâmicos, cujos campos mudam conforme o modelo de certificação adotado.

Referências

[1] Frakes, William; Kang, Kyo (2005). "Software Reuse Research: Status and Future". IEEE Transactions On Software Engineering, Vol.31, Nº 7, July.

[2] Bass, L. et al. (2000) "Market Assessment of Component-Based Software Engineering". SEI, Technical Report CMU/SEI-2001-TN-007.

[3] Wallnau, Kurt C (2004). "Software Component Certification: 10 Useful Distinctions". Technical Note, CMU/SEI-2004-TN-031.

[4] Underwriters Laboratories. http://www.ul.com. Acessado em 02/06/2007.

[5] ISO 9126 (2001). "Information Technology – Product Quality – Part1: Quality Model", International Standard ISO/IEC 9126, International Standard Organization.

[6] Bertoa, Manuel F.; Vallecillo, Antonio (2002). "Quality Attributes for COTS Components". Proceedings of the 6th International ECOOP Workshop on Quantitative Approaches in Object-Oriented Software Engineering, Spain.

[7] Alvaro, A.; Almeida, E. S.; Meira, S. R. L (2005). "Quality Attributes for a Component Quality Model". 10th International Workshop on Component-Oriented Programming (WCOP), Glasgow, Scotland.

[8] Carvallo, Juan Pablo; Franch, Xavier (2006). "Extending the ISO/IEC 9126-1 Quality Model with Non-Technical Factors for COTS Components Selection". International Workshop on Software Quality, May 21-21, Shanghai, China.

[9] B. Meyer (2003), "The Grand Challenge of Trusted Components", 25th International Conference on Software Engineering (ICSE), USA, pp. 660–667.

[10] Reusable Asset Specification. (2005). http://www.omg.org/docs/ptc/05-04-02.pdf. Acessado em 02/11/2005.

[11] Hoff, Arthur van; Partovi, Hadi; Thai, Tom (1997). "The Open Software Description Format (OSD)". Submission to the World Wide Web Consortium (W3C). http://www.w3.org/TR/NOTE-OSD. Acessado em 22/11/2006.

[12] Varadarajan, Sriram (2001). "ComponentXchange: An E-Exchange for Software Components". Dissertação de Mestrado, Indian Institute of Technology.

[13] Schuenck, Michael; Negócio, Yuri; Elias, Glêdson (2006). "Um Modelo de Metadados para Representação de Componentes de Software". VI Workshop de Desenvolvimento Baseado em Componentes.

[14] Oliveira, João Paulo; Schuenck, Michael; Elias, Glêdson. "ComponentForge: Um Framework Arquitetural para Desenvolvimento Distribuído Baseado em Componentes. VI Workshop de Desenvolvimento Baseado em Componentes, 2006.