1
Introdução

Todo projetista realiza, ao projetar algum artefato, uma seqüência de raciocínio e tomadas de decisão de projeto que resultam neste artefato. A captura do raciocínio utilizado pelo projetista durante a modelagem de uma aplicação raramente é realizada. Tal raciocínio é denominado na literatura como *Design Rationale* e, portanto, esse termo será utilizado ao longo desta dissertação com o intuito de manter a compatibilidade com documentos que tratam do mesmo assunto[1].

O *design rationale*, em toda sua abrangência, compreende não apenas os artefatos produzidos ao final do processo de desenvolvimento de software. Os artefatos finais representam a concretização das soluções adotadas pelo projetista, mas não explicitam as alternativas consideradas durante o seu desenvolvimento. O *design rationale* compreende todo o raciocínio utilizado ao longo da concepção dos artefatos, todas as questões e alternativas de soluções consideradas, além dos argumentos e justificativas para cada solução adotada ou rejeitada.

O rastreamento do *design rationale* é o ponto chave para a compreensão do *design* de um artefato, facilitando sua manutenção, seu reuso, sua evolução, a comunicação e colaboração entre diferentes desenvolvedores, e a resolução de problemas.

1.1.
Motivação

O conhecimento do *design rationale* de uma aplicação pode ser bastante útil para o desenvolvimento de outras aplicações baseadas no mesmo modelo formal e no mesmo domínio. Entretanto, registrar tal raciocínio durante o processo de *design* de um artefato pode se tornar um problema, caso não sejam fornecidas ferramentas adequadas ao projetista. O processo de registro do *design rationale* deve ser visto como parte do processo de construção do

[1] O mesmo se aplica ao termo *design*, que será mantido em seu original em inglês.

artefato. Documentar as decisões pode dificultar o processo de *design* caso seja visto como um processo separado da atividade de construir o artefato (Burge & Brown, 2000).

Por exemplo, se um projetista deseja modelar uma loja virtual para venda de livros utilizando a metodologia RUP[2], ele poderia consultar *design rationales* previamente registrados por outros projetistas e possivelmente adotar soluções já criadas para outras aplicações no domínio de lojas virtuais. Ou até mesmo utilizar-se de diferentes *design rationales* para montar o seu próprio raciocínio, baseado na união parcial dos raciocínios já desenvolvidos.

Utilizar-se de soluções previamente testadas e validadas gera um modelo mais confiável e em menor tempo; conciliar diferentes *design rationales* para um mesmo domínio pode gerar soluções mais completas. Desta forma este tipo de "reaproveitamento" de soluções poderia fazer com que fossem geradas aplicações mais completas e confiáveis, e em menor tempo.

O ideal seria a disponibilização de ambientes de desenvolvimento que permitissem a captura e posterior uso do *design rationale* utilizado. O projetista/desenvolvedor deve poder registrar seu raciocínio, com justificativas e argumentos que sustentem suas decisões de *design* e, conseqüentemente, as razões para a implementação dos artefatos que compõem a aplicação final. Sendo assim, o ambiente também deve disponibilizar uma lista de *design rationales* previamente criados para que o projetista/desenvolvedor possa avaliá-los e, se apropriado, reutilizar os *designs* na construção de sua aplicação.

1.2. Objetivos

A proposta deste trabalho é integrar a captura e eventual uso do raciocínio utilizado pelo projetista/desenvolvedor ao construir uma aplicação a uma ferramenta de apoio ao desenvolvimento de software.

O vocabulário da ontologia Kuaba, definido em Medeiros (2006), oferece um conjunto de primitivas que permitem o registro do *design rationale* capturado de forma precisa e estruturada. Com a integração da arquitetura conceitual do Kuaba a uma ferramenta de desenvolvimento, é possível capturar o *design rationale* de maneira semi-automática. Isto significa uma interação entre o desenvolvedor e a aplicação, de modo que as informações inseridas possam ser

[2] [Kruchten, 2001]

registradas e manipuladas a fim de rastrear o raciocínio utilizado no desenvolvimento da aplicação.

Para essa integração foi escolhida a ferramenta HyperDE (Nunes, 2005), conforme proposto na seção de trabalhos futuros de Medeiros (2006). O HyperDE é um ambiente de desenvolvimento e prototipação para aplicações hipermídia orientado a modelos.

Cabe aqui descrever os dois tipos de perfil mencionados: projetista e desenvolvedor. O projetista de software é responsável pela arquitetura da aplicação, ou seja, por elaborar soluções técnicas de implementação dos requisitos definidos para o software através de técnicas de modelagem, como por exemplo, orientação a objetos. O desenvolvedor é responsável pelo desenvolvimento de sistemas de software, tendo como principal atividade a codificação a partir dos requisitos e dos modelos do software definidos pelo projetista. No HyperDE, o usuário desempenha ambos os papéis, projetista e desenvolvedor, pois realiza etapas iterativas de construção e refinamento do modelo e do código da aplicação. Por esse motivo ao longo da dissertação os termos projetista e desenvolvedor serão utilizados para descrever o usuário do ambiente apresentado neste trabalho.

Este trabalho propõe a integração de parte do vocabulário da ontologia Kuaba ao ambiente HyperDE, resultando no ambiente denominado HyperDE+DR cujo projeto e implementação são apresentados nesta dissertação.

1.3. Estrutura da Dissertação

No capítulo a seguir são descritas as características principais de cada um dos trabalhos que serviram de inspiração para esta dissertação, visando o entendimento do projeto de integração desenvolvido.

No capítulo 3 é descrita a arquitetura do ambiente, com o detalhamento das extensões adicionadas e uma breve descrição das suas principais funcionalidades.

O projeto conceitual do ambiente é apresentado no capítulo 4, sendo descritos os requisitos, casos de uso e diagramas de classes, além da plataforma de implementação.

O capítulo 5 contém um exemplo de utilização do ambiente HyperDE+DR desenvolvido. São exemplificadas suas principais funcionalidades: a captura do

design rationale das aplicações construídas no ambiente e o uso de *designs* previamente registrados.

As contribuições e considerações finais são apresentadas no capítulo 6, juntamente com os trabalhos relacionados e as sugestões de trabalhos futuros. Também é apresentada nessa seção uma estimativa do esforço despendido para realizar as extensões propostas.

Ao final do texto há um glossário de termos utilizados ao longo da dissertação.

As referências bibliográficas que contribuíram para a elaboração deste trabalho são listadas ao final do documento.