A design pattern coupling role and component concepts: Application to medical software

Titre	A design pattern coupling role and component concepts: Application to medical software
Type de publication	Article de revue
Auteur	Fasquel, Jean-Baptiste [1], Moreau, Johan [2]
Editeur	Elsevier
Type	Article scientifique dans une revue à comité de lecture
Année	2011
Langue	Anglais
Date	2011/05
Numéro	5
Pagination	847 - 863
Volume	84
Titre de la revue	Journal of Systems and Software
ISSN	0164-1212
Mots-clés	Collaboration [3], Component [4], Design [5], Dynamic [6], medical [7], Role [8]
Résumé en anglais	One of the challenges in software development regards the appropriate coupling of separated code elements in order to correctly build initially expected high-level software functionalities. In this context, we address issues related to the dynamic composition of such code elements (i.e. how they are dynamically plugged together) as well as their collaboration (i.e. how they work together). We also consider the limitation of build-level dependencies, to avoid the entire re-compilation and re-deployment of a software when modifying it or integrating new functionalities. To solve these issues, we propose a new design pattern coupling role and component concepts and illustrate its relevance for medical software. Compared to most related works focusing on few role concepts while ignoring others, the proposed pattern integrates many role concepts as first-class entities, including in particular a refinement of the notion of collaboration. Another significant contribution of our proposal concerns the coupling of role and component concepts. Roles are related to the functional aspects of a target software program (composition and collaboration of functional units). Components correspond to the physical distribution of code elements with limited build-level dependencies. As illustrated in this paper, such a coupling enables to instantiate a software program using a generic main program together with a description file focusing on software functionalities only. Related code elements are transparently retrieved and composed at run-time before appropriately collaborating, regardless the specificity of their distribution over components.
URL de la notice	http://okina.univ-angers.fr/publications/ua1440 [9]
DOI	10.1016/j.jss.2011.01.026 [10]

Lien vers le document	http://dx.doi.org/10.1016/j.jss.2011.01.026 [10]

Liens
[1] http://okina.univ-angers.fr/j.fasquel/publications
[2] http://okina.univ-angers.fr/publications?f[author]=7736
[3] http://okina.univ-angers.fr/publications?f[keyword]=3462
[4] http://okina.univ-angers.fr/publications?f[keyword]=3463
[5] http://okina.univ-angers.fr/publications?f[keyword]=4376
[6] http://okina.univ-angers.fr/publications?f[keyword]=2512
[7] http://okina.univ-angers.fr/publications?f[keyword]=1862
[8] http://okina.univ-angers.fr/publications?f[keyword]=3467
[9] http://okina.univ-angers.fr/publications/ua1440
[10] http://dx.doi.org/10.1016/j.jss.2011.01.026