Advanced Science and Technology Letters
Vol.30 (ICCA 2013), pp.149-153
http://dx.doi.org/10.14257/astl.2013.30.32

ADS: Architectural Decision Supporter

Soojin Park[1], Seonghye Yoon[2], Deokyoon Ko[2]

[1]Graduate School of MOT, Sogang University, Seoul, South Korea
psjdream@sogang.ac.kr
[2]Dept.of Comp. Sci. & Eng. Sogang University Seoul, South Korea
seonghye@sogang.ac.kr, maniara@sogang.ac.kr

Abstract. Following the advancements in convergent devices, the trend of rapid increase in software complexity and the contrastingly shortened software product lifecycle have introduced new challenges to which the transformation from legacy single-core based system to multi-core based system have emerged. Unfortunately, software development process which provides adequate support for multi-core parallelization has not been around to keep up with the speed of advancements in multi-core based hardware systems. In this paper, to address this need, we propose a development process supporting transformation of existing single-core based software to a multi-core equivalent, and discuss the overall cost-saving potential in introducing automation tools for supporting applicable activities within the proposed process.

Keywords: multi-core parallelization, architecture authoring, architecture construction process.

1 Introduction

Through the emergence of multitudes of convergent digital devices, software development environment is evolving from developing software of fixed functionality for limited devices to developing software which supports tailoring of non-predefined features based on user needs. On the other hand, the lifecycle expectancy of a software product is quickly shortening. These environmental changes have led to the rise in demand for solutions to handle increased software complexity and development costs simultaneously. In response to this demand, multi-core based system with parallelization support has been proposed.

Although various hardware already provide multi-core capability, software developers are still faced with the repetitive and tedious challenge of rewriting existing single-core based software to support multi-core hardware and while maintaining satisfactory performance. For instance, we can consider a case where software product written for single-core hardware is modified to support quad-core hardware with desired clock frequency targeted to be 35% of its previous clock on single-core. In such case, it is difficult to accurately estimate the man-month required in achieving the desired output on a quad-core system, and no existing method sufficiently provides a solution.

This paper proposes an architecture-based multi-core transformation process as the basis of cost estimation. The process aims to transform single-core software currently performing satisfactorily on single-core hardware to support multi-core parallelism and deliver efficiently scaled performance in accordance to the increased number of cores. For this purpose, existing software architecture design and verification methodologies such as QAW [1], ADD [2], ATAM [3] and ARID [4] do not provide sufficient coverage for the entire spectrum of architecture construction phases, and further do not support process tailoring which we need in order to leverage domain knowledge of multi-core systems. To address these limitations, this paper proposes an architecture construction and evaluation process to serve as the guideline to resolve the shortcomings of existing methods, and propose Architectural Decision Supporter (ADS) as a tool to leverage domain knowledge in utilizing concurrency-related architecture patterns as to resolve problems encountered in transforming software towards multi-core based system. ADS is capable of suggesting verified and appropriately matched concurrency patterns to developers, providing reliable support for repetitive software rewriting tasks previously carried out without reliable reference or guidance.

The remainder of this paper is as follows: in Chapter 2, the software architecture construction process supporting transformation from single-core to multi-core support is detailed. Chapter 3 describes the scope and the means of automation support using ADS for activities defined in the proposed process, and Chapter 4 summarizes our conclusion.

2 Architecture Construction and Evaluation Process for Multi-core Based System

The proposed process is composed of 4 phases, as depicted in Figure 1. Respective activities in each phase are as follows:

- **Architectural requirements elicitation phase**: Accounting for the fact that most embedded software developers are likely to be resistant to documenting their designs, this phase starts from creating an activity diagram of the targeted single-core system through reverse-engineering of its source code. From the activity thread of the implementation model developed through source code analysis, use case specifications are drafted through inference. The drafted use case specifications are not black-box specifications based on the user perspective, but gray-box specifications [5] which include system internal implementation flows. After annotating whether if each step in the use case specifications requires concurrency, a quality attribute scenario regarding concurrency is created. By adding constraint descriptions related to architecture design, finally the software requirements specification (SRS) is completed. In the fortunate case where SRS for the legacy single-core system is already available, then most of activities in this phase can be skipped.

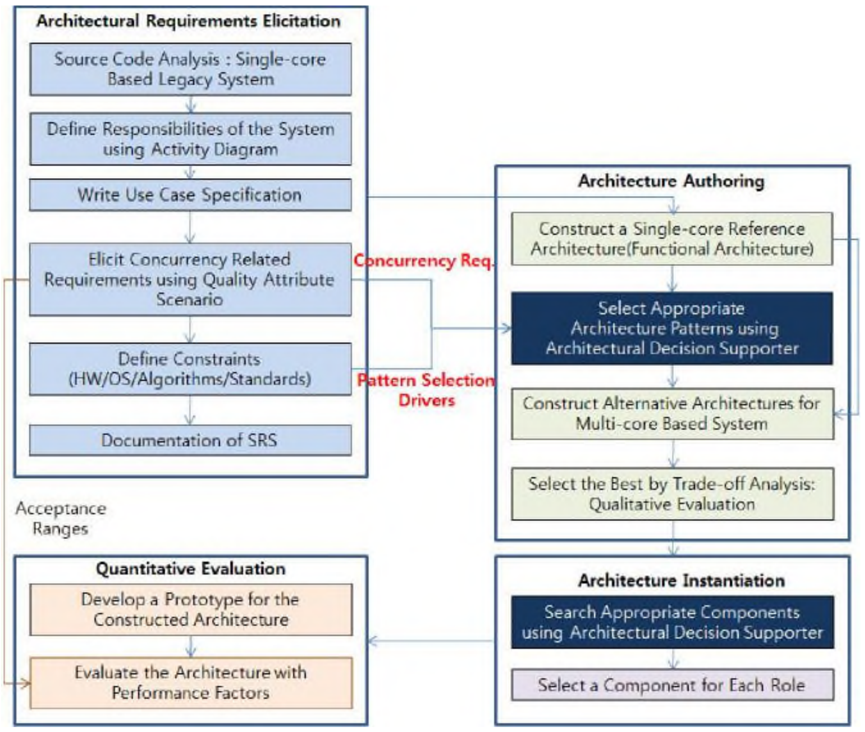
Architectural Requirements Elicitation
Source Code Analysis : Single-core Based Legacy System
Define Responsibilities of the System using Activity Diagram
Write Use Case Specification
Elicit Concurrency Related Requirements using Quality Attribute Scenario
Define Constraints (HW/OS/Algorithms/Standards)
Documentation of SRS
Concurrency Req.
Pattern Selection Drivers
Architecture Authoring
Construct a Single-core Reference Architecture(Functional Architecture)
Select Appropriate Architecture Patterns using Architectural Decision Supporter
Construct Alternative Architectures for Multi-core Based System
Select the Best by Trade-off Analysis: Qualitative Evaluation
Acceptance Ranges
Quantitative Evaluation
Develop a Prototype for the Constructed Architecture
Evaluate the Architecture with Performance Factors
Architecture Instantiation
Search Appropriate Components using Architectural Decision Supporter
Select a Component for Each Role

Fig. 1. An Architecture Construction and Evaluation Process for Transformation from Single-core Based System to Multi-core Based System

- Architecture authoring phase: In this phase we construct architecture for the legacy single-core system which satisfies all previous requirements in the use case specification. The resulting architecture will serve as reference for the forthcoming multi-core system architecture. When the reference architecture has been completed, then we select concurrency patterns from the patterns stored in ADS which both satisfies given concurrency requirements and does not violate design constraints, following a predefined rule. The selected concurrency patterns can then be applied to the reference architecture to enable development of multiple alternative architectures for the desired multi-core system. Among architecture candidates, the most appropriate one can be selected using ATAM’s qualitative evaluation.
- Architecture instantiation phase: In this phase, each component defined in our architecture is mapped to a component instance. While some components can be split or modified to resolve concurrency issues found during transformation towards multi-core, majority of components should be reflected back to system composition albeit at increased number of instances. Therefore, this phase can be viewed as a phase for constructing the system through selecting available components based on architecture definition.

- Quantitative evaluation phase: The most significant issue in transformation to multi-core system is how to achieve efficiently scaled performance proportionate to the number of increased cores. This complicates qualitative evaluation of ATAM in its appropriateness of assessing the soundness of a proposed architecture. In order to confirm if the actual performance increase satisfies predefined quality range, a prototype of the completed architecture is created to measure the approximate performance of each related attribute. If some required quality range is not satisfied, then the Architecture Authoring phase can be revisited for iteration.

3 ADS: Architectural Decision Supporter

ADS supports two activities (selecting appropriate architecture pattern and searching for appropriate constructed component) of the proposed process highlighted in Chapter 2. ADS provides three features for these activities: first is repository feature to store architecture-related knowledge of the developer’s organization, second is search feature to find appropriate architecture pattern, and the last is classification and search feature for reusable software components. ADS is provided to developers as an Eclipse plug-in, with Eclipse-like user interface identical intended to minimize tool learning time. All reusable artifacts, including concurrency-related patterns, are classified in conformance with Reusable Asset Specification (RAS) and therefore artifacts processed on ADA are compatible with all commercial tools supporting RAS.

4 Conclusion

This paper proposes a guideline for constructing software architecture for transmigration from single-core to multi-core hardware environment, under condition that performance should scale efficiently from single-core to proportionally matched performance relative to the increased number of cores. The proposed guideline makes contribution through providing rationale for selecting appropriate architecture patterns for multi-core system and measuring quantitative performance improvement. The effectiveness of ADS, however, remains to be further verified through additional case studies. Forthcoming research is to be further focused on applying ADS to real-life multi-core system architecture construction cases.

Acknowledgement. This research was supported by Next-Generation Information Computing Development Program through the National Research Foundation of Korea(NRF) funded by the Ministry of Science, ICT & Future Planning (2012M3C4A7033348)

References

1. M. Barbacci, R. Ellison, A. Lattanze, J. Stafford, C. Weinstock, and W. Wood, "Quality Attribute Workshops (QAWs), Third Edition," Software Engineering Institute, Carnegie Mellon University, Pittsburgh, Pennsylvania, Technical Report CMU/SEI-2003-TR-016, (2003)
2. R. Wojcik, F. Bachmann, L. Bass, P. Clements, P. Merson, R. Nord, and W. Wood, "Attribute-Driven Design (ADD), Version 2.0," Software Engineering Institute, Carnegie Mellon University, Pittsburgh, Pennsylvania, Technical Report CMU/SEI-2006-TR-023, (2006)
3. R. Kazman, M. Klein, M. Barbacci, T. Longstaff, H. Lipson, and S. Carriere, "The Architecture Tradeoff Analysis Method," Software Engineering Institute, Carnegie Mellon University, Pittsburgh, Pennsylvania, Technical Report CMU/SEI-98-TR-008, (1998)
4. P. Clements, "Active Reviews for Intermediate Designs," Software Engineering Institute, Carnegie Mellon University, Pittsburgh, Pennsylvania, Technical Note CMU/SEI-2000-TN-009, (2000)
5. S. Park, J. Byun and S. Park, “Applying Gray-Box based Software Requirements Specification Method to a Robot Patrolling System”, International Conference on Computer and Applications (2012) March 31; Seoul, Korea