Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 95 (2016) 48 – 57

Procedia Computer Science

Complex Adaptive Systems, Publication 6
Cihan H. Dagli, Editor in Chief
Conference Organized by Missouri University of Science and Technology
2016 - Los Angeles, CA

Executable Behavioral Modeling of System and Software Architecture Specifications to Inform Resourcing Decisions

Monica Farah-Stapleton[a]*, Mikhail Auguston[b], Kristin Giammarco[b]

[a] *PEO DHMS, 1700 N. Moore St, Rosslyn, VA, USA*
[b] *Naval Postgraduate School, Monterey, CA, USA*

Abstract

The size, cost, and slow rate of change of DoD Information Technology (IT) systems in comparison with commercial IT makes introduction of a new DoD system or capability challenging. Making design decisions without consideration of the whole system and its environment may result in unintended behaviors that have operational and financial impacts, often not visible until later testing. The complexity of these system interactions isn't cheap, impacting intellectual, programmatic, and organizational resources. Precise behavioral modeling offers a way to assess architectural design decisions prior to, during, and after implementation to mitigate the impacts of complexity, but in and of itself does not lead to estimates of the effort and the cost of those design decisions. This research introduces a methodology to extract Unadjusted Function Point (UFP) counts from architectural behavioral models utilizing a framework called Monterey Phoenix (MP), lightweight formal methods, and high level pseudocode for use in cost estimation models such as COCOMO II. Additionally, integration test estimates are informed by extracts of MP model event traces. These unambiguous, executable architecture models and their views can be inspected and revised, in order to facilitate communication with stakeholders, reduce the potential for software failure, and lower costs in implementation.

Peer-review under responsibility of scientific committee of Missouri University of Science and Technology

Keywords: architecture; lightweight formal methods; behaviors; event traces; function point analysis; COCOMO II, complexity.

* Corresponding author. Tel.: 571-830-0507
E-mail address: monica.f.farahstapleton.civ@mail.mil

Peer-review under responsibility of scientific committee of Missouri University of Science and Technology
doi:10.1016/j.procs.2016.09.292

1. Introduction

This paper presents a domain independent methodology, hereafter referred to as ThreeMetrics to extract Unadjusted Function Point (UFP) counts from discrete architectural behavioral models, created from the Monterey Phoenix (MP)[1] modeling language and framework, for use in cost estimation models such as COnstructive COst MOdel II (COCOMO® II)[2,3]. The MP model itself is a rich source of additional information including scenarios (use cases) that can be extracted from the MP model to inform distinct integration test case development, as well as views of instances of the architecture model that can be inspected for accuracy and facilitate communication with stakeholders. The UFP count, event traces, and views were the inspiration for the name ThreeMetrics.

1.1. Background

Historically, there have been significant but often disconnected efforts to develop architectural descriptions of new and legacy systems. Architectural design and analysis are powerful mechanisms that allow the capture of design decisions early in the design process, so that it can be assessed and modified without incurring unnecessary costs of incorrect implementations. Unfortunately, architectural design decisions are often captured on a system by system basis, using a spectrum of representations from natural language to formal notations. These inconsistent systems architectures are then analyzed through manually intensive methods such as inspections and reviews. System and software architecture and development efforts are often unrelated, incomplete, or duplicative, with a technically and programmatically unsustainable result. This is an unfortunate state of affairs because architectures matter. "Every system has an architecture, whether or not it is documented and understood[4]." Not only is the architecture of a software system complex, but so are the programmatic, organizational, and resourcing constructs that interact with each other throughout the software lifecycle. All these architectures deserve the attention of technical and programmatic decision makers because if constructed properly, they can not only capture design decisions but also inform resourcing decisions and reduce the complexity of sociotechnical implementation. The ThreeMetrics methodology applies elements of the Function Point counting process to MP architecture models, in order to extract an Unadjusted Function Point count from MP models, and inform technical and programmatic decision making.

1.2. Monterey Phoenix (MP)

MP is a behavioral model for system and software architecture specification based on event traces, and supports several architecture composition operations and views. As an executable architecture model, it can be used to automatically generate examples of the behaviors (e.g. use cases) for early system architecture testing. This software and system modeling framework can also be used to capture design decisions such as precedence, inclusion, concurrency, and ordering (dependency relation between activities).[5,6,7] MP's foundation is in lightweight formal methods, which plays a key role in assessing the complex behaviors of a software intensive system, and in the development of formal specifications for the system and the environment. Formal methods are essential to behavioral modeling of complex systems, because they remove ambiguity from architectural modeling. As with all assessments, lightweight formal methods based architectural assessments are assisted by visual representations and automated tools. Such tools provide immediate feedback, assist in identifying errors once an early architecture draft is constructed, and allow the user to reason about the model. There are many tools supporting lightweight formal methods based analysis, including the MP Analyzer on Firebird[8], Eagle6[9], and Alloy Analyzer[10]. Firebird and Eagle6 are implementations of the MP Framework. Eagle6 is a commercial tool, which has been graciously made available for select research purposes. Firebird is an NPS implementation that is publically available, and was ultimately selected for this work.

1.3. Function Point Counting

The ThreeMetrics methodology leverages the International Function Point User Group (IFPUG) Function Point counting method defined in Function Point Counting Practices Manual Release 4.3.1. "A Function Point is a normalized metric used to evaluate software deliverables and to measure size based on well-defined functional characteristics of the software system."[11] The unit of functional size for this method is called a Function Point (FP).