Addressing Performance Requirements Using a Goal and Scenario-Oriented Approach

Zhiming Cai[1] and Eric Yu[2]

[1] Department of Computer Science, University of Toronto, Canada
zmcai@cs.toronto.edu
[2] Faculty of Information Studies, University of Toronto, Canada
yu@fis.utoronto.ca

Abstract. Performance requirements should be addressed as early as possible during requirements analysis and architectural design. This paper presents a goal-oriented and scenario-oriented approach for qualitatively addressing and refining performance requirements. The goal-oriented language GRL[1] is used to represent the refinement of performance goals from abstract to concrete ones, eventually operationalizing them into design alternatives. The Use Case Maps (UCM)[4] notation is used to represent system operations at a high level of abstraction using scenarios.

1 Introduction

Performance requirements and performance evaluation can have a global impact on alternatives for target system [2]. The goal of performance modelling is to address performance requirements and make the performance of system more predictable. Performance modelling of the whole system at the requirements analysis and high-level design stages can provide feedback into the decision-making process prior to detailed design and implementation.

Group Communication Server (GCS) system described in [3] is a multi-user system for document management. An approach called PERFECT in [3] quantitatively evaluates the different proposals for concurrency architecture by "virtual implementations". An important challenge is to qualitatively address performance of different architectures without implementation. Quantitative metrics and qualitative treatments are complementary for performance modelling.

This paper proposes an approach "PeGU" for performance modelling by using GRL (Goal oriented Requirement Language) and UCM (Use Case Map). The modeler uses PeGU to qualitatively evaluate each of architectures over each possible technology choice decision at an early stage. PeGU treats performance requirements as softgoals and goals of GRL. GRL is a modelling language proposed at University of Toronto for supporting goal-oriented modelling and reasoning of requirements. Modelling both goals and scenarios is complementary and goal-scenario coupling may aid in identifying further goals, additional scenarios and scenario steps. Use Case

A. Banks Pidduck et al. (Eds.): CAISE 2002, LNCS 2348, pp. 706-710, 2002.

Maps is used for representation of scenarios. The structure and the behavior can be described in the same UCM diagram. The paths of execution are represented against the background of the system components.

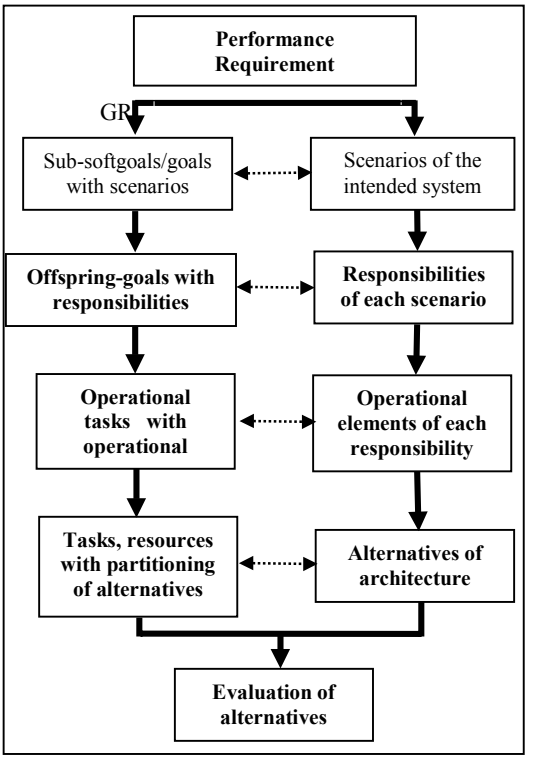

Fig.1. Overview of PeGU

2 Overview of PeGU

2.1 High-Level Performance Requirement

The high-level performance requirements are qualitatively specified as GRL softgoals and goals. The UCM and GRL modeling method are employed alternately in the following steps. (see Fig.1.)

2.2 Scenario

UCM side – The scenarios of the intended system are outlined in large-grained behaviour pattern according to system requirements;

GRL side – The high-level performance softgoals and goals are distributed to sub-softgoals (goals) for different scenarios.

2.3 Responsibility and Offspring-Softgoals/Goals

UCM side – Along each scenario path, some localized actions which the system must perform are viewed as responsibilities. With analyzing the paths of execution for the scenarios, responsibilities are figured out for each path;

GRL side – Each performance requirement softgoal/goal is decomposed into offspring, correlating with the responsibilities of each scenario.

2.4 Operationalization into Task

UCM side – The responsibilities of each scenario are refined into operational elements, which specify concrete notation in the target system;

GRL side – The performance softgoals/goals are refined into operational tasks. The different operations on UCM side are related to different tasks on GRL side.

2.5 Architecture Design

Alternatives for system architecture are designed in different possible ways. The partitioning of the alternatives is indicated by GRL tasks and resources.

2.6 Evaluation

Evaluation for each alternative generally starts from leaf of the bottom layer and then up through higher layers in GRL. The impact of decision on each operation of each alternative is evaluated and propagated to above elements. Evaluation may result change and optimizing of architecture design.

We will use the example of Group Communications Server (GCS) in [3] to illustrate how PeGU models and evaluates performance requirements qualitatively and deals with trade-offs among a number of factors.

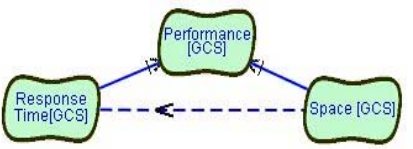

Fig.2. High-level performance requirements of GCS

3 PeGU Methodology - Applying to GCS

1. We define *Performance[GCS]* softgoal as a top level requirement of GCS. The contributing factors to overall performance are response time and space usage. *Performance[GCS]* is thus refined into two sub-softgoals: *ResponseTime[GCS]* and *Space[GCS]* as shown in Fig.2. in GRL graphical notation. *ResponseTime[…]* means seeking decreased or low response time for specific user requests; *Space[…]* means decreased or low space usage. Here, *Performance[GCS]* and its subgoals are related by an *AND* "contribution" link.

The *Space[GCS]* has correlation-link with *ResponseTime[GCS]* because the relationship between ResponseTime and Space is not an explicit desire, but is a side-effect.

2. Fig.3 shows scenarios of GCS in UCM notation: *Update Document, Notification, Send New Document, Subscription, UnSubscription* and *Get New Document*. The responsibilities *upd, noti, new, sub, uns* and *get* specify the actions for each scenario. The timestamp points *Tarr, Tupd, Tnot, Tnew, Tsub, Tuns* and *Tget,* represent the start and end points for response-time requirement of each scenario.

   The high-level performance softgoals in GRL are distributed to sub-softgoals on the scenarios, as in Fig.4. *ReponseTime[upd]* Softgoal means good response time for scenario *update*. Other softgoals are similar for other scenarios.

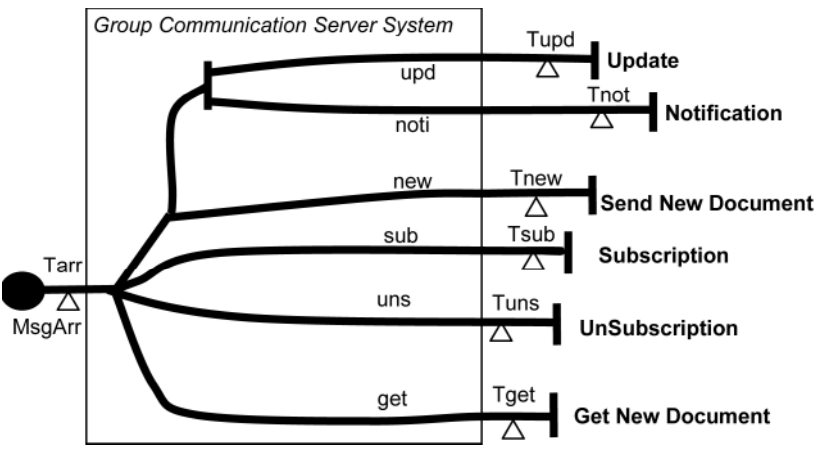

Fig.3. Scenarios of GCS

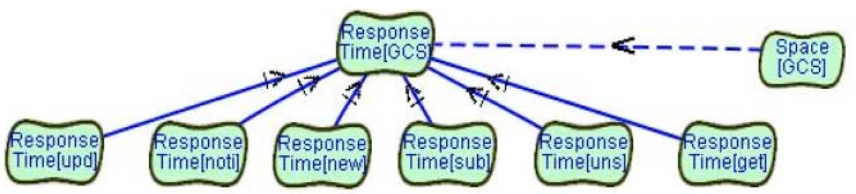

Fig.4. GCS performance softgoals on the scenarios

3. The execution of each path can be considered as several segments or activities in UCM. Some concrete behaviours can be extracted from the abstract responsibilities. The performance requirement for each scenario can be refined further in GRL, correlating with segments and activities of each path in UCM.
4. As the operational elements for the responsibilities of each scenario become concrete in UCM, the operationalization of each performance softgoal and goal can be made in GRL. The tasks and resources are exploited and connected to the softgoals.

5. After the concrete operational elements are worked out both in UCM and GRL, the different architectural design options can be considered. The partitions of each alternative are coupled with related tasks for different softgoals.
6. Fig.5 shows the softgoal refinement after (3)(4)(5) and evaluation of response-time for the concurrent architecture of "Thread-per-disk" in GRL notation. The labels of the softgoals connected to labeled tasks are propagated from bottom to up. For instance, *ResponseTime[upd]* is undecided as *ResponseTime[PreMsgUpd]* is undecided though *ResponseTime[WriteDisk(upd)]* is satisfied, but with And-contribution. *ResponseTime[GCS]* softgoal is eventually considered as undecided since response-time on most (4/6) scenarios are undecided after the propagating. The designs and evaluations for other architectures can be done in a similar way.

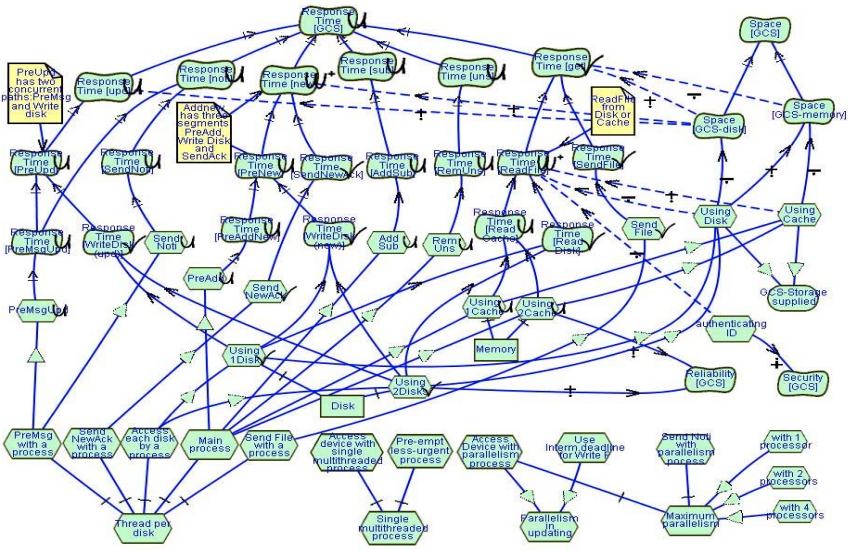

Fig.5. Refinement of performance softgoals and evaluation for architecture of "Thread-per-disk"

References

1. GRL On-line at: http://www.cs.toronto.edu/km/GRL/
2. Nixon, B.A., Management of Performance Requirements for Information System. *IEEE Transactions on SE*, 26(12): 1122-1146,2000
3. Scratchley, W.C. and Woodside, C.M., Evaluating Concurrency Options in Software Specifications. *Seventh International Symposium on Modelling, Analysis and Simulation*, College Park Maryland, USA, 1999.
4. UCM On-line at: http://www.usecasemaps.org/pub/UCMtutorial/