Cloud Adoption: A Goal-Oriented Requirements Engineering Approach

Shehnila Zardari
Friday April 8, 2011
ABSTRACT

Cloud services are often delivered as black-boxes, which means that the understanding of cloud services features is frequently partial and uncertain on the part of the acquirer. The selection of Cloud Service Provider is one of the most important activities taking place in the context of moving the user operations to the cloud. It involves the assessment of how well cloud services satisfy user requirements. Due to the nature of cloud, mismatches may occur between what is required from the cloud (i.e. user requirements) and what the cloud is able to provide (i.e. its features). In addition, these mismatches may result in a number of risks, such as insufficient adherence to the requirements by the Cloud Service Provider, low confidence in Cloud Service Provider’s quality and unwanted cloud features. We argue that successful selection of a suitable cloud depends on the effective analysis of mismatches and management of risks. We motivate the need for a new requirements engineering methodology for systematically helping businesses and users to adopt cloud services and for mitigating risks in such transition. The methodology is grounded in goal oriented approaches for requirements engineering. We argue that Goal Oriented Requirements Engineering (GORE) is a promising paradigm to adopt for goals that are generic and flexible statements of users’ requirements, which could be refined, elaborated, negotiated, mitigated for risks and analysed for economics considerations. The methodology can be used by small to large scale organisations to inform crucial decisions related to cloud adoption.
CONTENTS

1 INTRODUCTION..........................................................................................................................4
2 MOTIVATION..............................................................................................................................6
3 CONTRIBUTION.........................................................................................................................7
4 RELATED WORK ..........................................................................................................................8
  4.1 GORE for Functional Requirements .......................................................................................8
  4.2 GORE for Non-Functional Requirements ...............................................................................8
  4.3 i*/ Tropos ..............................................................................................................................9
  4.4 KAOS......................................................................................................................................10
  4.5 Some Other GORE Approaches ...........................................................................................10
5 EXISTING RESEARCH IN CLOUD COMPUTING ....................................................................10
6 EVALUATION PLAN...................................................................................................................11
7 TIMELINE FOR PRODUCTION OF THESIS PROPOSAL ......................................................12
8 REFERENCES...............................................................................................................................13

APPENDICES

Appendix A

Systematic Analysis of Risks in Cloud Architecture

Appendix B

Cloud Adoption: A Goal Oriented Requirements Engineering Approach
1 INTRODUCTION

The ever increasing need for data processing, storage, elastic and unbounded scale of computing infrastructure has provided great thrust for shifting the data and computing operations to the cloud. IBM advocates cloud computing as a cost efficient model for service provision [1]. The adoption of cloud computing is gaining momentum because most of the services provided by the cloud are low cost and readily available. The pay-as-you-go structure of the cloud is particularly suited to Small and Medium Enterprise (SME) who have little or no resources for IT services [5].

The growing trend of cloud computing has led many organisations and even individuals moving their computing operations, data, and/or commissioning their e-services to the cloud. Moving to the cloud has reduced the cost of computing and operations due to resource sharing, virtualization, less maintenance cost, lower IT infrastructure cost, lower software cost, expertise utilization and sharing etc [3]. For example, the New York Times managed to convert 4TB of scanned images containing 11 million articles into PDF files, which took 24 hours for conversion and used 100 Amazon EC2 Instances [4]. Such relatively quick conversion would be very expensive if done in-house. The term cloud computing may simply refer to different applications over the Internet or the hardware shared between different users [2]. Buyya et al have defined cloud as:

“A Cloud is a type of parallel and distributed system consisting of a collection of inter-connected and virtualized computers that are dynamically provisioned and presented as one or more unified computing resources based on service level agreements established through negotiation between the service provider and consumer” [9]

In a cloud, hardware/software are shared and utilized as services at lower cost. Many services are now offered in the realm of cloud computing. These are:

- Infrastructure as a service (IaaS),
- Platform as a service (PaaS),
- Software as a service (SaaS).

**Infrastructure as a Service:** Infrastructure as a Service is a model in which an organization outsources the equipment required to perform operations like storage, hardware, servers etc.
Cloud Service provider provides all the hardware needed for operations and is responsible for maintaining it. The client pays for what he uses. Amazon’s Elastic Compute is the example of such a service.

**Platform as a Service:** Cloud Service Provider provides a platform to the user on which a user can develop an application. The applications are delivered to the users through cloud service provider’s infrastructure. Coghead and Google App Engine are the examples of PaaS.

**Software as a Service:** SaaS delivers a single application through the browser to thousands of users. Users are not required to invest on purchasing servers or software licensing. Payment is made on the basis of the data transferred and some fixed rent. Google App Engine is a representative example of SaaS.

In cloud context, a service level agreement (SLA) is expected to mediate consumers’ expectations with respect to cloud service provision. Despite the fact that businesses have already started to exploit the potentials of the cloud as a paradigm for services, there is a total absence for foundation and methodologies for informing the process of cloud adoption. As a result, the current practice for adopting clouds (and their services) could be described as ad hoc, lacking rigour and systematic guidance. Selection is often biased by the reputation of the cloud provider, their SLA statements, terms, conditions and promises, recommendations and clouds’ reputations, past experiences and the like from biased or subjective ad hoc inputs. In particular, there is a general lack of requirements engineering methodologies, which could be suited for cloud adoption. The objective of our research is to develop a requirements engineering framework, which could help users screen, match, and negotiate their requirements against cloud services’ provision. The framework will also assist in the problem of managing the tradeoffs associated with matches and mismatches of users’ requirements against cloud service provisions.

To bridge the gap, practitioners are very likely to beg, borrow, and steal from requirements engineering in-the-small to benefit the ultra-large-scale paradigm as it is the case of the cloud. Though the fundamentals of engineering requirements in both paradigms exhibit resemblance, requirements engineering for the cloud requires novel flexible and scalable approach, which address the interplay between technical, economics-driven considerations and shifts requirements engineering towards a utility-based engineering for
consumers’ satisfaction for cloud service provisions such as software as-, infrastructure-, data storage-, and/or platform as- services.

2 MOTIVATION

We looked at a case study of a leading cloud service provider- referred to as Indus throughout this report. The case study revealed that there are many risks associated with cloud adoption. Since the cloud is perceived as a “black box”, a user has little or no control over the promises set by cloud’s SLAs. For instance, the user cannot negotiate the SLAs with the cloud service provider and hence has to agree with the set terms and conditions.

As an example, Indus sufficiently describes their security mechanisms in the agreement accompanying their SLAs “Indus Web Services Customer Agreement.” Interestingly enough, Indus mentions that the nature of communication over the Internet is unpredictable and largely insecure. Given the vulnerability of the Internet, Indus cannot guarantee the security of user’s content. While Indus strives for a secure environment, the security responsibility and accountability lie solely on the users and the organisation using the services. In the event of any breach in security requirements, Indus is not entirely liable to the user for any unauthorized access, use, deletion, corruption or destruction of user’s content. The service provider has attempted to win the confidence and trust of the users by publishing the agreement, yet it has failed to ascertain the individual needs of different users. Indus has a shared responsibility environment for the safety of user’s content, where one of the inherent problems with such “joint” responsibility is that it makes accountability difficult. Referring to SLAs terms and conditions, Indus absolves itself of any responsibility in the wake of anything goes wrong. For instance, Indus recommends customers to encrypt data transferred over the network. There are tradeoffs involved with large data encrypted over the network: this will lead to higher processing time, which might affect cloud performance and consequently violate the promises set in the SLA. Despite the promised elasticity of cloud computing, resources continue to be scarce. E.g. enhancing security provision may cause performance bottleneck. As a result, the promised Quality of Service (OoS) can’t be often met as the SLA terms and conditions stipulate.
We call for a novel requirements engineering methodology for cloud adoption, which could assist businesses in screening, selecting cloud service providers and negotiating their services and qualities of provision. The framework aims at helping businesses screen, match, and negotiate their requirements against cloud services’ provision. The framework will also assist in the problem of managing the tradeoffs associated with matches and mismatches of users’ requirements against cloud’s provision. Such provision aims at objectively evaluating the strategic decisions, satisfaction of the technical and operational goals, cost and value of such decisions and the tradeoffs involved in moving to the cloud. Despite rapid growth of cloud use, there is a general lack of systematic methodologies aiming at such. Decisions regarding the selection of cloud service providers are made on ad hoc basis based on recommendations or on the reputation of the service provider. The lack of such methodologies exposes businesses considering the cloud for unpredictable risks. It would be expensive to get “locked in” with a wrong cloud. Evaluating pre adoption choices at early stages is cost-effective strategy to mitigate risks of probable losses due to wrong or unjustified selection decisions. Furthermore, the framework aims at assisting users in assessing their requirements against cloud provision.

Due to the dynamic nature of the cloud, mismatches may occur between what is required by the user and what is provided by the cloud provider. The framework will assess the suitability of cloud service providers by exploring mismatches, managing risks and suggesting possible tradeoffs. We use goal-oriented approach for cloud provision. The expected beneficiaries of the work are small to large businesses, educational institutes and even individuals, who wish to exploit the cloud. Such work is novel and bridges an important gap in making the process of cloud adoption more transparent, systematic and user oriented. It would be worth noting that ongoing research on cloud selection has addressed the problem of dynamic selection of the cloud services with respect to QoS. Up to our knowledge, no research has looked at cloud adoption from a requirements perspective.

3 CONTRIBUTION

The overall objective of our research is to investigate new processes, strategies and mechanisms to support requirements engineering process for cloud adoption. We argue that by better supporting the pre-adoption requirements process; it is possible to improve the
Cloud Adoption: A Goal-Oriented Requirements Engineering Approach

quality of decisions made during the selection of Cloud Service Provider. The benefits of improving the quality of decisions are:

a. increasing the general satisfaction of stakeholders;

b. reducing the risks of an unsuitable cloud to be selected;

c. controlling the costs involved in the selection process.

In order to address these issues, we propose a novel methodology which could assist users in selecting and screening the cloud services based on users requirements. We propose an approach that involves requirements engineering process for cloud adoption based on goal-oriented requirements engineering. We choose GORE to model user requirements as goal refinements offer the right level of abstraction to involve decision makers for validating choices that have been made and for suggesting alternatives. The expected contribution of our thesis is

i. Modelling goals i.e. user requirements and tradeoffs analysis for satisfying those goals.

ii. Modelling obstacles in achieving goals.

iii. A life cycle for the process of cloud adoption.

4 RELATED WORK

A comprehensive use of Goal-oriented requirements engineering can be found in [6]. Requirements are represented in the form of goals in GORE. GORE approaches have normally the following activities: goal elicitation, goal refinement and various type of goal analysis, and finally the assignment of goals to the agents. The advantage of using this approach is that a goal graph provides vertical traceability from high-level strategic concerns to low-level technical details; it allows evolving versions of the system under consideration to be integrated as alternatives into one single framework. Goal-oriented approaches have received significant attention over the years, where goals were used for modeling different kind of requirements.

4.1 GORE for Functional Requirements

GORE can be used to model functional requirements. Functional requirements specify the functions or services of the system. Goals may be referred as functional concerns or quality
attributes [7]. Functional goals result in functional requirements e.g. Satisfaction Goals are concerned with the satisfaction of users’ requirements [16]. A functional goal typically defines maximal set of desired scenarios. In a GORE process, goals are used to make operational models like use cases, activity diagrams etc.

4.2 GORE for Non-Functional Requirements

Non-functional requirements (NFRs) represent software system qualities (e.g. ease of use, maintainability, etc.) or properties of the system as a whole. NFRs are generally more difficult to express in an objective and measurable way. Thus, their analysis is more difficult. GORE has also been used to model the non-functional requirements [24]. The Non-Functional Requirements Framework was proposed in [15] and was later developed in [33]. The NFR framework focuses on modeling and analyzing the non-functional requirements. The framework deals with the following activities: acquiring NFRs for any system, decomposing NFRs, identifying design alternatives for meeting NFRs, selecting the way to operationalize NFRs, and evaluating the decisions. NFRs are represented in the form of softgoals. There are no precise criteria for satisfaction of a softgoal. For softgoals one needs to find solutions that satisfy the goal to a sufficient degree. Softgoals can rarely be fully satisfied. Softgoals can be refined using AND/OR refinements with obvious meaning. Mylopoulos et al used goal oriented approach for eliciting, specifying and refining the non-functional requirements [15]. The softgoal interdependency graph (SIG) is the main tool for modeling the non-functional requirements. The graph can represent softgoals and softgoals refinement (AND/OR). As softgoals are being refined, a stage is reached when the goals are sufficiently detailed and cannot be refined further. If a softgoal receives contribution from a number of other softgoals, then the results of contribution of every single offspring are combined to get the overall contribution for satisfying the parent goal.

4.3 i*/Tropos

i* [34] is an agent-oriented modeling framework that can be used for requirements engineering, business process modeling and redesign [29], organizational impact analysis, and software process modeling. Considering the fact that i* supports the modeling activities that take place before the system requirements are formulated, it can be used for both early and late requirements engineering phases. The i* framework models the environment of the
system-to-be in the early requirements phase. The framework assists in the analysis of the domain by diagrammatically representing the stakeholders of the system, their goals, and their relationships. The $i^*$ models are used to suggest the new system configurations and the new processes and assess them to know how well they meet the functional and non-functional requirements of the users in the late requirements engineering phase. $i^*$ is based on the idea of intentional actor and intentional dependency. Actors can be either agents or roles. Agents are systems or humans, with specific abilities. A role is an “abstract actor embodying expectations and responsibilities” [35]. Dependencies between actors are intentional if they appear as a result of agents following their goals.

4.4 KAOS

KAOS is a goal-oriented requirements engineering approach having a set of formal analysis techniques. KAOS stands for Knowledge Acquistion in automated Specification [36] or Keep All Objects Satisfied [37]. KAOS is a framework which has different levels of expressions and reasoning: semi formal to model goals, qualitative for selection of alternatives, and formal whenever required for accurate reasoning [37].

Goal definition in KAOS is “prescriptive statement of intent about some system whose satisfaction in general requires the cooperation of some of the agents forming that system” [38] Goals in KAOS may be indicative of services i.e. functional goals or quality of service (QoS) i.e. non-functional goals. Goals are refined with AND/OR abstraction hierarchies. Goals refinements ends when goals cannot be further refined or when every subgoal is realizable and can be assigned to an agent.

4.5 Some Other GORE Approaches

[18] demonstrates another use of Goal-oriented approaches in Agent Oriented Programming (AOP) for open architecture that need to change and evolve due to changing requirements. GORE was also used to model the system architecture to meet changing business goals and for evolving systems [19, 20]. One interesting application of GORE, which has inspired our work is that of [21, 22, 23], where GORE was used to inform the process of selecting Commercial off the Shelf (COTS) products matching user's requirements. Though the fundamental use of GORE exhibits resemblance with that of [21, 22, 23] the problem of
Cloud adoption is by far more challenging as we are dealing with "open loop" environments, with dynamic, unbounded and elastic scale where continuous service evolution is the norm.

5 EXISTING RESEARCH IN CLOUD COMPUTING

Research efforts over the years have looked at the problem of service discovery with runtime mechanisms to inform and optimize the selection e.g. self-managed applications in the cloud [25] and self-optimizing architectures [10] etc. Privacy in the cloud [31, 32] has got significant attention by the researchers. Up to our knowledge, there has been no research on cloud procurement and adoption from requirements engineering perspective. The need for such research is timely as there is complete lack of systematic methodologies, which could help stakeholders screen, match, negotiate their requirements against cloud services’ provision. Such helps in managing the tradeoffs associated with matches/mismatches of users’ requirements against cloud provision and mitigating likely risks. There has been a recent research on Cloud Migration [30] but this research lacks any systematic approach for refining and elaborating users’ requirements. Their research is concerned with the enterprises decision to migrate services to the cloud informed by cost benefit analysis. Cloud Security Alliance has published a paper online concerning with cloud assessment before adoption called CSC Cloud Adoption Assessment [39]. The CSC Cloud Adoption Assessment has nothing regarding the tradeoffs involved with satisfying different requirements.

6 EVALUATION PLAN

We plan to evaluate our methodology by modeling a case study. We aim to experiment its applicability on small and medium enterprises’ case study. Users’ requirements can be mapped to the available benchmarks through individual forums or cloud service providers’ SLAs. Such experiment shall result in a comparison of strengths, weaknesses and limitation of our approach when applied to different cases.

We aim to verify that:

i. Our approach is cost effective.

ii. User requirements regarding cloud selection have been met.

iii. The risks associated with cloud adoption have been minimized.
7 TIMELINE FOR PRODUCTION OF THESIS PROPOSAL

I intend to submit thesis proposal by the end of August 2011, as per following roadmap.

<table>
<thead>
<tr>
<th>S. No.</th>
<th>Date</th>
<th>Activity/Task</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>April, 2011</td>
<td>Model case study of Indus</td>
</tr>
<tr>
<td>2</td>
<td>May, 2011</td>
<td>Analyse the obstacles and conflicts among goals.</td>
</tr>
<tr>
<td>3</td>
<td>June, 2011</td>
<td>Decide upon an approach for cost/benefit analysis of our methodology</td>
</tr>
<tr>
<td>4</td>
<td>July, 2011</td>
<td>Write thesis proposal</td>
</tr>
<tr>
<td>5</td>
<td>August, 2011</td>
<td>Write a research paper for 34th International Conference on Software Engineering</td>
</tr>
</tbody>
</table>
8 REFERENCES

Cloud Adoption: A Goal-Oriented Requirements Engineering Approach


