InCLOUDer: A Formalised Decision Support Modelling Approach to Migrate Applications to Cloud Environments

Adrián Juan-Verdejo [1,2], *Steffen Zschaler* [3], *Bholanathsingh Surajbali* [2], *Henning Baars* [1], *Hans-Georg Kemper* [1]

[1] *Information Systems Chair, Stuttgart University, Germany. adrian.juan@cas.de,{baars,kemper}@wi.uni-stuttgart.de*

[2] *CAS Software A.G, Karlsruhe, Germany. {adrian.juan, b.surajbali}@cas.de*

[3] *Department of Informatics, King's College London, UK. szschaler@acm.org*

***Abstract*—An increasing number of organisations want to migrate their existing applications to cloud environments to benefit from the increased scalability, flexibility, and cost reduction. Additionally, systems migrated to cloud environments have to fulfil their functional requirements, satisfy their users' requirements, and meet the organisation's criteria for cloud migration. All these different dimensions driving the migration decision conflict with each other. Therefore, organisations trade them off for one another. Migrating applications to cloud environments becomes a complex decision process for which organisations need assistance. We provide a decision support system to assist organisations by taking into account the formal description of the parameters affecting the cloud migration and our proposed metrics for objective and subjective criteria. Our approach to cloud migration allows organisations to describe their cloud migration criteria; the architecture, properties, and requirements of their applications; and the available cloud service offerings. We semi-automate the migration decision with our transparent formalisations to quantify criteria and constraints.**

***Keywords*-multi-cloud migration; decision support**

I. Introduction

Nowadays, many organisations and individuals develop their computing solutions directly for cloud environments to profit from the potential cost reduction and increase in agility [1]. Often organisations also want to leverage cloud computing to run applications which have been already developed for a different computing environment [1]. Those organisations have to properly adapt their applications to the right cloud environment in order to avoid the risk of failing in this migration process [2]. Many interdependent dimensions [3] and parameters [2] affect the decision of to what cloud service offering to migrate an application and how to do it. Hence, the migration decision becomes a multi-criteria optimisation problem.

The existing research works usually address some specific parts of this decision-making problem such as the cloud environment selection [4] or the effects of a specific dimension affecting the adaptation and migration of applications to cloud environments [3]. They focus either on the technical, organizational, social, economic, business, or sensitivity concerns that affect applications cloud migration [3], [5], [6]. The problem with these approaches is that they separate aspects of the migration decision that are connected and depend on one another. Whether to migrate parts of an application and how to do it depends on the organisation's interest and focus in moving their applications to cloud environments, the application's architecture and properties, and the selected cloud service [7]. Additionally, stakeholders describe differently and not transparently their migration criteria, their applications' requirements, and the cloud service properties. This hinders the quantification and comparison of the different options to migrate an application.

In this work, we aim to provide a formal description of the parameters that affect the migration of applications to cloud environments. We propose InCLOUDer, our approach to cloud migration that allows organisations to describe the target application's architecture, requirements, and properties; the available cloud service offerings; and their criteria for application migration. We propose a transparent formalisation to quantify these criteria and the constraints limiting the cloud migration process. Which, in turn, partially automates the migration decision. We show exemplary criteria and constraints. We have validated our approach by developing a prototype to support organisations in solving the multi-criteria optimisation problem.

In the next Section, we motivate our work. Next, in Section III, we describe research works related to ours, highlight our contribution in comparison with them, and explain the research gaps we fill. Next, in Section IV, we formalise the migration problem our research work tackles. Further, we explain our InCLOUDer cloud migration decision support system in Section V and how it semi-automatically migrates applications in Section VI. Section VII describes the prototype we developed to support the theoretical problem formalisation and the InCLOUDer cloud migration decision support system. Finally, we conclude in Section VIII.

II. Motivation

The migration of applications to cloud environments entails benefits, but also concerns varying from technical, security and trust issues to organizational, and legal constraints. Applications to be migrated consist of several components which communicate with each other to comply to the application's functional and non-functional requirements. If organisations do

not correctly adapt their applications to the target cloud environment, they might experience problems to meet some quality or economic requirements. Typical adaptation issues range from compatibility issues between cloud environments and migrated applications to restrictive licenses that forbid organisations moving proprietary software components. Sometimes, performance could erode due to the increase in latency due to the wide-area communication introduced when an application component is moved to a cloud environment whereas another dependent component is kept locally to meet sensitivity-related requirements.

Organisations have different interdependent criteria and constraints to move their applications to cloud environments. Organisations typically trade criteria with one another. Improving the system in a dimension, as increased computing power, could entail damaging the system in another one, say cost. These dimensions are not always equally important to all organisations and sometimes there are some constraints that an organisation cannot overcome. As an example, an organisation might not be allowed to move health care data outside some private premises and has to run part of the computation on the local premises in addition to the cloud environment. The nature of the application components to be migrated, the application's users, and the interest of the organisation migrating the application shape how to adapt that application to cloud environments and to what specific cloud service offering to migrate.

We have evidenced the need to assist organisations in the complex decision-making problem of migrating their applications to cloud environments. This is our motivation to help organisations in selecting the cloud service offering suitable for their application and their migration criteria. Organisations may benefit from our approach as it releases them from the burden of measuring and considering all interdependent dimensions, criteria, constraints, and factors that affect the cloud migration and adaptation decision.

III. Related Work

The migration of existing applications to cloud environments demands from organisations to select a suitable target cloud service [8], [4] and adapt their applications to them according to their needs and the specifics of their application [2]. Whether to migrate or not an application, how to do it, and to which cloud service form a multi-dimensional decision problem affected by a lot of factors, sub-factors, and parameters [1], [2]. Organisations have to evaluate many conflicting criteria before making any decision and the problem becomes a multiple criteria decision making problem [9].

The rapid growth of cloud service offerings together with the heterogeneous cloud providers business models and their inability to transparently describe their services hinders the selection of the most suitable target cloud service [2]. Some research works classify cloud services for selection by taking into account many factors and attributes that evidence a cloud service's performance, cost and features [4] [10] while others measure the performance-related parameters to compare the options [11]. Unlike our work, these research works decide how to migrate an application without including information about the target application characteristics nor the motivations and criteria of a specific organisation to migrate to a cloud offering. Some research works such as [4], [12] use the Analytical Hierarchy Process to compare service providers and rank them but they do not automate the process and therefore require significant user input which hinders their usage for real-case scenarios.

Some other works incorporate organisation's criteria into their decision support tools or prototypes to assist the migration of applications to cloud environments, but they only focus in a specific concern and do not consider the others [3]. Nevertheless, the criteria are interconnected and oppose one another. Organisations have to take into account all criteria together instead of in isolation before making the migration decision [13]. We seek inspiration in research works which deal with the migration of applications to cloud environment by taking into account one specific dimension of the process such as the business and economic implications [14], technical-related challenges [13], organisational implications [3], or security and privacy [6], [5]. We differ from those research works in how we use the criteria-specific knowledge and how we put those criteria together to consider how the cloud service selection affects the criteria evaluation and how some criteria conflict with others. We find inspiration in their works but formalise our own criteria to judge the different options to migrate an application.

IV. Problem Formalisation

In this Section, we formalise our cloud migration problem starting with the application App to be migrated to the selected cloud service offerings (CSO). We then formalise the organisation's requirements $Reqs$ to migrate App. Finally, we formulate our cloud migration problem as finding the optimal split of App to both selected CSO and the premises local to the organisation that maximises the gains in the selected cloud migration criteria experienced by the migrated application. The maximisation problem takes into account the organisation's criteria ko_j and constraints co_j

for cloud migration.

$$App = (\ C, R \subseteq C \times C, \overrightarrow{P_C}, \overrightarrow{P_R}\ ) \tag{1}$$

We describe in (1) the application App as a composition of several data and computing **components** C which perform their tasks and communicate with one another through their **relations** R, $R \subseteq C \times C$. A relation R represents the link between two components that enables them to transfer data to each other. Application App's components C possess several intrinsic **properties** $\overrightarrow{P_C}$, $P_{Ci} : C \rightarrow \mathbb{R}$; as do the relations (edges) between those components $\overrightarrow{P_R}$, $P_{Ri} : R \rightarrow \mathbb{R}$. The number of cores a component needs to operate or the memory allocation it needs are examples of components' properties $\overrightarrow{P_C}$. As for the relations' properties $\overrightarrow{P_R}$, an example is the data transfer rate two components need in order to provide a satisfactory quality of service.

$$Reqs = (K, W, CO\ ) \tag{2}$$

We formulate in (2) the **requirements** $Reqs$ defined by the organisation which wants to migrate App to cloud environments. The set of cloud migration criteria K are aggregated according to matrix W to rank only the migration alternatives which comply to the set of constraints CO. K comprises either objective or subjective criteria. Our approach considers an objective criterion, ko_j as seen in (4), if there is a metric to measure a migration alternative for that specific criterion or a subjective criterion ks_q, based on human intervention, if there is not. The organisation pairwise compares all criterion in K to one another to obtain matrix W, which describes the relative importance of all criteria. From matrix W we obtain the weights eigen vector $\vec{w}$ used in our maximisation problem. We explain this in more detail in Section V.

$$re\text{-}scatter :\ C \rightarrow (\ \perp \cup\ CSO\ ) \tag{3}$$

We describe in (3) the *re-scatter* function due to its importance for the definition of the organisation's criteria and constraints. Based on application App's components C, the *re-scatter* function returns the application components C **re-scattered** into both the local premises $\perp$ and the ***CSOs***. Examples of CSOs are a *small Amazon Elastic Compute Cloud (EC2) instance*, a *big Amazon EC2 instance*, or an *extra small Azure instance* provided by cloud providers such as *Amazon*, or *Azure*. A cloud service provider specifies its ***CSO's* properties** P_{CSO_i} where $\overrightarrow{P_{CSO}} : (\perp \cup\ CSO) \rightarrow \mathbb{R}$. An example of a P_{CSO_i} is the amount of memory provided by a CSO such as a *small Amazon EC2 instance*.

We consider two kinds of **criteria**, which are relevant for the organisation which wants to move application App to cloud environments, depending on whether these are objective ko_j or subjective ks_q. We mathematically express in (4) a **objective migration criterion**. ko_j incorporates knowledge related to the available CSOs, the needs and characteristics of application App, and the organisation's cloud migration criteria to move application App to cloud environments.

$$ko_j :\ \overbrace{(\ C \rightarrow (\ \perp \cup\ CSO\ )\ )}^{re\text{-}scatter} \times 2^C \times 2^R \times \overbrace{(C \rightarrow \mathbb{R})^m}^{\overrightarrow{P_C}} \times \underbrace{\times (R \rightarrow \mathbb{R})^n}_{\overrightarrow{P_R}} \times \underbrace{((\perp \cup\ CSO) \rightarrow \mathbb{R})^p}_{\overrightarrow{P_{CSO}}} \rightarrow \mathbb{R} \tag{4}$$

Each criterion ko_j represents a dimension related to the cloud migration—e.g. cost, performance, or sensitivity—and we measure it for every non-rejected migration strategy. That is, a migration alternative to re-engineer application App by re-scattering its components to both the organisation's premises and the CSO. Non-rejected migration alternatives comply to the constraints defined (see 5). ko_j depends on the function to *re-scatter* application components, $C \rightarrow (\perp \cup CSO)$; the application components and the relations between them, C and R; the m properties of the components, $\overrightarrow{P_C}$; the n properties of the relations between components, $\overrightarrow{P_R}$; and the p properties $\overrightarrow{P_{CSO}}$ of the CSO. In Section VI-B, we show several examples of criteria.

Only if we have not specified a metric for a criterion, we use a **subjective migration criterion** ks_q calculated by requesting organisations to pairwise compare all migration strategies for each of these criteria. An example of a subjective criterion is the trust an organisation has in a specific CSO if no metric has been defined for it. How we calculate ks_q is described by Saaty and Vargas [15] and it evaluates to a number.

$$co_g :\ \overbrace{(\ C \rightarrow (\ \perp \cup\ CSO\ )\ )}^{re\text{-}scatter} \times 2^C \times 2^R \times \overbrace{(C \rightarrow \mathbb{R})^w}^{\overrightarrow{P_C}} \times \underbrace{\times (R \rightarrow \mathbb{R})^x}_{\overrightarrow{P_R}} \times \underbrace{((\perp \cup\ CSO) \rightarrow \mathbb{R})^y}_{\overrightarrow{P_{CSO}}} \rightarrow Boolean \tag{5}$$

As for the set of **constraints**, CO in (2); we define in (5) each element of the set as a constraint co_g which limits the acceptable migration alternative's properties. All the potential strategies to migrate App have to either comply with all constraints in CO or be disregarded. We formalise every co_g in resemblance with the criteria formalisation in (4). (5) defines when to reject a migration alternative according to the organisation's criteria, application App architecture and its properties, and the CSO. Its output is a *Boolean* value to show whether a constraint is respected or

not. As an example, by using this constraint formula, an organisation can define a constraint stating that a Java Virtual Machine (JVM) has to be runnable for a particular re-engineered version of application *App* deployed into a *CSO*. More examples of these constraints are given in Section VI-A.

We look for the optimal split of application *App*'s components *C* done by the *re-scatter* function (3), so that we maximise (6):

$$max \sum_{i=0}^{|K|-1} w_i \cdot K_i(re\text{-}scatter, C, R, \overrightarrow{P_C}, \overrightarrow{P_R}, \overrightarrow{P_{CSO}}) \quad (6)$$

where w_i refers to the eigen value of the relative weight of criterion i within the weights eigen vector $\vec{w}$ from matrix W in (2); and K_i to an objective or subjective criterion i as described by ko_j and ks_q. Our maximisation problem, in (6), is subject to the satisfaction of all constraints $co \in CO$.

V. The InCLOUDer Cloud Migration Decision Support System

The aim of our cloud migration decision support system is to rank the different alternatives to adapt an application to migrate it to cloud environments. InCLOUDer discards unsuitable alternatives according to the specified constraints—as formalised with (5) in Section IV and illustrated with examples in Section VI-A—and suggests the highest-ranked strategy. Once not suitable cloud migration alternatives are discarded, InCLOUDer needs to calculate local rankings for every alternative based on the migration criteria. These criteria depend on many properties and sub-properties related to: the application, the *CSO*s, and the user requirements. Hence, it is a multi-dimensional decision problem [9] that needs a technique to evaluate the structured hierarchy of migration criteria. The hierarchy of criteria makes it easier for organisations to weight different criterion to one another and also complies with the natural nature of the migration criteria. Each criterion impacts the ranking process in accordance with how determinant it is for the success of the application migration. A criterion can be further specialised by defining its sub-criteria and usually conflict with other criteria. Weighting so many conflicting criteria while taking into account the interdependence between them burdens organisations and might disrupt their judgement. Additionally, some attributes cannot be easily measured and quantified to a numerical value. Even though we can propose metrics to quantify subjective criterion, we expect that sometimes it will not be feasible to quantify all criteria and we will need assistance of the organisation. Therefore we enabled InCLOUDer to gather information from organisations so that our decision support system can calculate these subjective criteria through pairwise comparing cloud migration alternatives. In order to easily consider trade-offs and structured relationships between objective and subjective criteria, quantify them, and aggregate them; we propose InCLOUDer which includes a ranking system based on the Analytical Hierarchy Process to address the multi-criteria decision making [15]. In usual simple multi-criteria decision making problems, all criteria are expressed in the same unit but our process lets us weight criteria expressed in different dimensions. Weighting the cost of a migration strategy—expressed in euros— against the respect to national regulations—expressed in Boolean rationale—is an example of the kind of challenges the Analytical Hierarchy Process helps with.

InCLOUDer applies five steps to find the optimal cloud migration strategy: Step 1, modelling the problem as a hierarchy with a goal, criteria, and cloud migration alternatives; Step 2, prioritising the criteria; Step 3, evaluating the different alternatives for every criterion; Step 4, checking the consistency of the judgements; and finally Step 5, coming to a decision.

A. InCLOUDer steps

Firstly in **Step 1**, we describe our multi-criteria problem with a three-layered **hierarchy model**—suitable for the application of the Analytic Hierarchy Process—by specifying the goal, criteria, and alternatives of our decision problem. The goal is finding the highest-ranked cloud migration alternative which splits the application components into both the organisation's local premises and *CSO*s. The criteria are the different dimensions an organisation considers when moving an application to cloud environments. For the sake of simplicity, we defined *Reqs* in (2) within Section IV on the basis of the criteria and their weights; but we use the Analytical Hierarchy Process approach for hierarchically structured criteria. As in [15], we calculate the decision matrices *DM* with the organisation's cloud migration criteria and weights. *DM* directly depends on a set of criteria within a level of the structured hierarchy of criteria, these criteria relative weights, and the evaluation of each cloud migration alternative per each of these criteria [15]. We describe the decision matrix by formulating each DM_y. That is, the decision matrix for a level y within the structured hierarchy of criteria consisting of M alternatives and N migration criteria. InCLOUDer calculates each criterion Cr_l , where $l = [0..N]$, for each leaf and branch node within each criteria layer y of the hierarchy. For each criterion in Cr_l without sub-criteria, or leaf node, we calculate Cr_l by computing either an objective ko_j

or a subjective ks_q criterion. For branch nodes—that is criteria with sub-criteria— within criteria layer y, we calculate Cr_{lower}, where $lower = [0..Q]$, which is a criterion similar to Cr_l but in a lower level of the criteria hierarchy.

Table I: Decision Matrix

$$DM_y = \begin{array}{c|ccccc} \boldsymbol{Cr_l}: & Cr_1 & Cr_2 & Cr_3 & ... & Cr_N \\ \boldsymbol{w_l}: & w_1 & w_2 & w_3 & ... & w_N \\ \hline \boldsymbol{A_1}: & a_{11} & a_{12} & a_{13} & ... & a_{1N} \\ \boldsymbol{A_2}: & a_{21} & a_{22} & a_{23} & ... & a_{2N} \\ . & . & . & . & . & . \\ \boldsymbol{A_z}: & a_{z1} & a_{z2} & a_{z3} & \boldsymbol{a_{zl}} & a_{zN} \\ . & . & . & . & . & . \\ \boldsymbol{A_M}: & a_{M1} & a_{M2} & a_{M3} & ... & a_{MN} \end{array}$$

As seen in Table I, each w_l from the eigen vector [15] for all criteria Cr_l in layer y is a normalised weight value per criterion l; A_z refers to each alternative to adapt application App to a CSO; and each $a_{z,l}$ evaluates each alternative A_z in terms of the decision criteria Cr_l and the relative importance (or weight) of each criterion w_l. Each $a_{z,l}$ evaluates directly to a value for every Cr_l without sub-criteria or uses another decision matrix DM_{lower} for the calculation. We describe our taxonomy of criteria and some exemplary criteria in Section VI-B. InCLOUDer relies on criteria templates as pre-defined criteria sets and pairwise comparisons between each criterion. Organisations customize or create their own templates tailored to their specific migration scenario; or otherwise re-use criteria templates applicable to their specific cloud migration scenario. A criteria template serves organisations that have similar priorities in how to move to cloud environments applications which posses similar requirements. Finally, the bottommost level contains the alternatives, which capture how to adapt the application and which components to re-scatter to the organisation's local premises and to a particular CSO. Our approach to cloud migration generates the alternatives subject to the defined constraints CO.

A criteria template contains the pairwise comparisons of each criterion to one another so that InCLOUDer calculates the eigen vector with all criteria weights, that is **Step 2** or the **criteria prioritisation**.

In Section VI, we explain with more detail **Step 3** that is, the **alternatives evaluation** for all criteria. Then, InCLOUDer aggregates the migration alternatives by taking into account the eigen vector of weights computed in Step 2 from the $\vec{W}$ matrix.

Next in **Step 4**, we **validate that the consistency ratio**, CR, does not trespass the acceptance level of 10% that Saaty and Vargas proposed and thoroughly explained [15]. We check that pairwise comparisons derive from consistent or near consistent matrices by dividing our consistency index by the random index $CR = CI/RI$. CR assess the uncertainty of the decision made.

Finally in **Step 5**, InCLOUDer ranks all alternatives and suggests a **decision** on cloud migration. The organisation in charge of the migration could still pick another one as our approach allows organisation to correct the criteria and their weights, the application components model and properties, the cloud environment model, or the formula InCLOUDer uses to evaluate an alternative for a particular criterion. The organisation can re-define the latter by either changing how an alternative is automatically evaluated or manually evaluating that alternative.

VI. InCLOUDer's Semi-Automatic Cloud Migration

The semi-automation of the migration decision relieves organisations which want to move an application to cloud environments of the burden of considering all options and the multi-dimensional repercussions of picking one over another. InCLOUDer automatically generates the alternatives, rejects the non-viable ones, and finally weights them for every criterion. We first show in Section VI-A examples of constraints co_g to describe the automatic alternatives generation subject to the constraints in CO. Then, we explain in Section VI-B the criteria by showing examples which explain how InCLOUDer automatically weights the remaining alternatives.

A. Alternatives Generation Subject to Constraints in CO

InCLOUDer generates architectural design alternatives to migrate an application to cloud environments subject to the constraints defined by an organisation to support the hierarchy modelling or Step 1 described in Section V-A. Every alternative specifies which components run within the local premises of the organisation C_{local} and which components C_{cso} migrate to a specific CSO. Where C_{cso} denotes a set of components migrated to a specific CSO:

$$C_{cso} \subseteq C : \forall c \in C_{cso} \quad re\text{-}scatter(c) \in CSO \tag{7}$$

As seen in (5) in Section IV, a constraint limits the acceptance of cloud migration alternatives subject to a specific property. A constraint represents a requirement vital for an application regardless of whether it runs within the local premises or in a selected cloud environment. An organisation adds, extends, or removes constraints with our cloud migration decision support system. We show two examples of constraints included in InCLOUDer:

Security and privacy constraint: InCLOUDer copes with sensitivity issues by keeping locally components

which store or process data that organisations do not want to move out of their premises to a particular cloud service. This can be due to lack of trust in that cloud provider or in the communications channel used for migration. In this case, InCLOUDer keeps locally the components with high sensitivity levels; that is, *High Sensitivity* ($P_{C_{sensitivity}} = 2$) and *National Security High Sensitivity* ($P_{C_{sensitivity}} = 3$) [5]. Where $P_{C_{sensitivity}}$ is an InCLOUDer P_{Ci} component property, as described in (1) within Section IV, whose value represents a component's sensitivity level. As we show in (8), InCLOUDer returns a $co_{sensitivity}$ value equal to *False* for cloud migration alternatives which do not comply with the sensitivity-related constraints and are therefore rejected:

$$\begin{aligned} co_{sensitivity}(\textit{re-scatter},\ C, R, \overrightarrow{P_C}, \overrightarrow{P_R}, \overrightarrow{P_{CSO}}) = \\ = \neg\, \exists c \in C_{cso} :\ [\textit{re-scatter}(c) \in CSO \\ \wedge\ P_{c_{sensitivity}} \geq 2] \end{aligned} \tag{8}$$

An organisation which trusts the migration mechanisms and the CSO would relax this constraint.

Explicit requirements: InCLOUDer generates alternatives which comply with the application *App* requirements according to the components model and the components' properties specified by an organisation. Additionally, InCLOUDer lets organisations explicitly define constraints disallowing the migration of a component to a CSO which does not supply a vital feature. For example, an explicit constraint does not allow the generation of an alternative that migrates a component to a CSO which does not support a required storage technology or proprietary tool, say *MongoDB* as in constraint $co_{mongoDB}$ in (9).

$$\begin{aligned} co_{mongoDB}(\textit{re-scatter},\ C, R, \overrightarrow{P_C}, \overrightarrow{P_R}, \overrightarrow{P_{CSO}}) = \\ = \neg\, \exists c \in C_{cso} :\ [\textit{re-scatter}(c) \in CSO \\ \wedge\ P_{c_{mongoDB}} = 1 \wedge P_{CSO_{mongoDB}} = 1] \end{aligned} \tag{9}$$

where $P_{c_{mongoDB}}$ is a component property P_{Ci} as seen in 1 within Section IV whose value 1 represents a component which requires *MongoDB*; and where $P_{CSO_{mongoDB}}$ is a component property P_{CSOi} as seen in Section IV whose value 1 represents a CSO which supports *MongoDB*.

B. Automatic Alternatives Weighting

The automatic alternatives weighting pertains to Step 3 described in Section V-A. Provided that we have defined a metric for a criterion, InCLOUDer automatically calculates the score of all alternatives for that criterion. Next, it aggregates the alternatives based on the criteria eigen vector, as described in Step 2 or the criteria prioritisation Step, to obtain the global ranking for all the alternatives.

Our cloud migration decision support system uses our taxonomy and criteria metrics to measure a cloud migration alternative for a specific criterion. We present an extensible criteria taxonomy based on the Service Measurement Index framework [10] and SMICloud [4]:

1) ***Accountability****:* is a subjective attribute ko_{ac} that affects trust and is built based on several other attributes such as: auditability, compliance, contracting experience, data ownership, ease of doing business, provider ethics, or service sustainability [10]. For our problem, accountability includes the commitment of a cloud provider to comply with policies and legal, ethical, and moral obligations. The degree of accountability of a cloud provider depends on the mechanisms they put into action to measure, prevent, and act to take responsibility for the stewardship of personal and confidential data entrusted to them [10]. InCLOUDer currently considers the alternative accountability equal to the CSO's accountability if the alternative migrates a component C with considerable sensitivity level. That is, all levels except *Low*—$P_{C_{sensitivity}} = 0$. Hence, the following sensitivity levels: *Moderate*—$P_{C_{sensitivity}} = 1$, *High*—$P_{C_{sensitivity}} = 2$, and *National Security High*—$P_{C_{sensitivity}} = 3$. We formalise this criterion as $ko_{ac} = P_{CSO_{ac}}$ if $\exists c \in C_{CSO} : P_{C_{sensitivity}} > 0$.

2) ***Agility****:* the use of cloud environments to run an application enables an organisation to quickly and cost-efficiently expand, change, and integrate new IT capabilities to accommodate business needs. Agility depends on its sub-criteria of elasticity, portability, adaptability, and flexibility. We show as an example how to calculate the degree of portability ko_{port} of an alternative to a CSO. Hence, the degree to which the effort of porting the application to the new environment ko_{cPort} is less than the cost of redevelopment ko_{cReDev} or $1 - \frac{cost\ to\ port}{cost\ to\ redevelop}$ [16]:

$$\begin{aligned} k_{ac}(\textit{re-scatter},\ C, R, \overrightarrow{P_C}, \overrightarrow{P_R}, \overrightarrow{P_{CSO}}) = \\ = 1 - \frac{k_{cPort}(\textit{re-scatter},\ C, R, \overrightarrow{P_C}, \overrightarrow{P_R}, \overrightarrow{P_{CSO}})}{k_{cReDev}(\textit{re-scatter},\ C, R, \overrightarrow{P_C}, \overrightarrow{P_R}, \overrightarrow{P_{CSO}})} \end{aligned} \tag{10}$$

Currently InCLOUDer relies on human intervention to determine with pairwise comparisons ko_{cPort} and ko_{cReDev} as explained in the cost criteria Section VI-B4.

3) ***Assurance****:* defines the probability of the application to comply to the Service Level Agreement (SLA) and depends on the services' availability, maintainability, recoverability, reliability, resiliency or fault tolerance, service stability, or serviceability [10]. These sub-criteria are highly affected by the local premises and the selected CSO properties running the applica-

tion's components. For example, in our approach the average **availability**—hence, service uptime divided by the total service time— depends on the availability properties specified with InCLOUDer for each component, the SLA of the selected CSO, and the local infrastructure:

$$ko_{av}(\textit{re-scatter}, C, R, \overrightarrow{P_C}, \overrightarrow{P_R}, \overrightarrow{P_{CSO}}) = \\ = \sum_{c \in C} \frac{P_{CSO_{up}}}{P_{CSO_{up}} + (P_{CSO_{down}} + (P_{c_{down}} - P_{\perp_{down}}))} \tag{11}$$

where $P_{CSO_{up}}$ and $P_{CSO_{down}}$ are P_{CSO_i} infrastructure properties that represent the uptime and downtime in milliseconds of a component running either on the local infrastructure ($\perp$) or a CSO; and $P_{c_{down}}$ is the total downtime time of a component which are down if the infrastructure that hosts it is down, given that $P_{c_{down}} \geq P_{CSO_{down}}$.

4) ***Cost****:* organisations are interested in lowering the effort to adapt the application to cloud environments ko_{eff}, the on-going cost of running the application ko_{og}, and the one-time migration costs ko_{mc} [1].

Given that $P_{c_{size}}$ is a component property P_{Ci} whose value represents a component's size; the C_{cso} set formulated in (7); and the $C_{(cso,\, P_{c_{size}}=i)}$ set which refers to a set of components in which each component's size is i, $P_{c_{size}} = i$, given that $C_{(cso,\, P_{c_{size}}=i)} \subseteq C_{CSO} \subseteq C$ as formalised in (12):

$$C_{(cso,\, P_{c_{size}}=i)} = \{c \in C_{cso} : \quad P_{c_{size}} = i\} \tag{12}$$

We formulate the one-time migration cost criterion ko_{mc} in (13) as an example of a cost criterion:

$$ko_{mc}(\textit{re-scatter}, C, R, \overrightarrow{P_C}, \overrightarrow{P_R}, \overrightarrow{P_{CSO}}) = \\ = \sum_{i=0}^{N} M^i * |C_{(cso,\, P_{c_{size}}=i)}| \tag{13}$$

where a sum runs over the different component sizes i from 0 to N, which InCLOUDer defines from *Very Small* ($P_{c_{size}} = 0$) to *Very Large* ($P_{c_{size}} = 6 = N$); and M is the difference in magnitude from one level, such as *Very Small*, to the next one, *Small*.

Our current approach to measure the effort to adapt ko_{eff} an application to cloud environments takes into account the cost of porting ko_{cPort} or redeveloping the application ko_{cReDev}. Currently we incorporate organisations into the cloud migration decision-making by letting them pairwise compare alternatives to calculate these criteria. Our future work will calculate ko_{eff} without human intervention.

5) ***Performance****:* is a complex research field on its own therefore InCLOUDer will, as future work, integrate existing approaches to estimate performance to measure the performance effects of re-scattering applications' components such as the Palladio[1], Maude[2] , or Kieker approaches[3].

6) ***Security and Privacy****:* concern the majority of organisations moving to cloud environments because they fear losing control of parts of the applications and data they host outside their premises [5]. As seen in Section VI-A, InCLOUDer defines a constraint that forces leaving *High* and *National Security High* Sensitivity components locally to the organisation. Our approach considers irrelevant, in terms of security and privacy, whether to move *Low Sensitivity* data out of an organisation's premises due to their lower sensitivity level [5]. Which leaves InCLOUDer with judging whether to migrate data and components marked with *Moderate Sensitivity* to a CSO. The fewer elements with *Moderate Sensitivity*—that is $P_{c_{sensitivity}} = 1$—an alternative migrates to a CSO, the higher InCLOUDer ranks that alternative:

$$ko_{sen}(\textit{re-scatter}, C, R, \overrightarrow{P_C}, \overrightarrow{P_R}, \overrightarrow{P_{CSO}}) = \\ = |C_{(cso,\, P_{c_{sensitivity}}=1)}| \tag{14}$$

where we consider the C_{cso} set as seen in (7); and the $C_{(cso,\, P_{c_{sensitivity}}=1)}$ set of components in which each component migrated to that CSO has a *Moderate* sensitivity level. We formulate $C_{(cso,\, P_{c_{sensitivity}}=1)} \subseteq C_{CSO} \subseteq C$ as:

$$C_{(cso,\, P_{c_{sensitivity}}=1)} = \{c \in C_{cso} : \quad P_{c_{sensitivity}} = 1\} \tag{15}$$

Given that $P_{C_{sensitivity}}$ is a component property P_{Ci} whose value represents a component's sensitivity level, (14) returns the number of components with *Moderate Sensitivity* moved to a cloud environment.

7) ***Usability****:* the ease of use of a component depends on its accessibility, client requirements to use it, installability, learnability, operability, suitability, transparency, and understandability [10]. These sub-criteria are rather imprecise and subjective. Therefore, InCLOUDer allows organisations to subjectively estimate usability based on the application's components model and some CSO properties related to the ease of use.

VII. Prototype

With the aim of validating our theoretical formulation and the InCLOUDer cloud migration decision support system we built a prototype to find and rank adaptations to migrate applications to cloud environments. The InCLOUDer supplies three EMF-based[4] editors to let organisations model the application for migration, the migration criteria, and the CSO which

[1]PCM: *https://sdqweb.ipd.kit.edu/wiki/Palladio_Component_Model*
[2]Maude: *http://maude.cs.uiuc.edu*
[3]Kieker: *http://kieker-monitoring.net/framework/*
[4]Eclipse Modelling Framework: *www.eclipse.org/modeling/emf/*

provide InCLOUDer with the input it needs as described in Section IV. One of these editors lets organisations model the relations R between components C as well as their properties, $\overrightarrow{P_C}$ and $\overrightarrow{P_R}$. InCLOUDer gathers the cloud migration criteria and their importance relative to each other—ko_i, ks_q, and W—with another editor. InCLOUDer includes a cloud service modelling approach based on variability points [17] to let cloud providers model their CSO by describing CSO's properties and capabilities (CSO and $\overrightarrow{P_{CSO}}$).

Given these three models, InCLOUDer generates suitable alternatives (Section VI-A) and automatically weights them to present the highest-ranked migration strategy (Section VI-B). InCLOUDer contains two feedback mechanisms to both improve the migration support and let the organisation drive the decision-making regardless of the migration strategy suggested.

VIII. Conclusion

We have explained the multi-dimensional decision-making process carried out to migrate applications to cloud environments and how to formalise its effects in the cloud migration criteria. With the aim of helping organisations cope with these effects we supply the InCLOUDer cloud migration decision support system which builds on top of the Analytic Hierarchy Process. InCLOUDer allows formalising the cloud migration problem by defining the cloud migration criteria, the cloud service offerings, and the architecture and properties of the application. Given all that, InCLOUDer assists organisations in adapting their applications to cloud environments according to their many interdependent criteria for cloud migration. We provided a taxonomy of these organisation's objective and subjective criteria for cloud migration related to accountability, agility, assurance, cost, performance, security and privacy, and usability. We provided examples of these criteria and explain how InCLOUDer follows the Analytical Hierarchy Process approach to trade a criterion off for others and how it only generates alternatives which comply to our exemplary constraints. An alternative for cloud migration re-scatters applications' components to both cloud environments and the local premises to cope with sensitivity issues. Our decision support system weights these alternatives by taking into account our criterion metrics and their importance relative to other criterion. When no metric is provided the weighting is done manually by the organisation.

In this paper, we have provided a formalisation approach for multi-criteria application migration to cloud environments. As future work we aim to enhance and evaluate our approach with automated metrics. In particular, we will focus on the migration of Business Intelligence (BI) applications because BI is an interesting study due to the large number of interconnected components. Through this exercise we plan to evaluate and quantify the gains accomplished in terms of the cloud migration criteria presented in this paper. We also plan to test the incorporation of multiple cloud environments and how to leverage them on runtime instead of on design time as we do today.

Acknowledgment

This work is part of the RELATE project supported by the European Commission under the 7th Framework Programme FP7 with Grant agr. no. 264840ITN.

References

[1] F. A. Armbrust, M. and R. Griffith, "A view of Cloud Computing," *Communications of the ACM*, 2010.

[2] V. Andrikopoulos, T. Binz, F. Leymann, and S. Strauch, "How to adapt applications for the cloud environment: Challenges and solutions in migrating applications to the cloud," *Computing*, vol. 95, no. 6, pp. 493–535, 2013.

[3] A. Khajeh-Hosseini, I. Sommerville, J. Bogaerts, and P. Teregowda, "Decision support tools for Cloud migration in the enterprise," in *IEEE Conf. on Cloud*, 2011.

[4] S. K. Garg, S. Versteeg, and R. Buyya, "Smicloud: A framework for comparing and ranking cloud services," in *2011 4th IEEE Int. Conf on UCC*, 2011.

[5] D. Rosado, R. Gomez, D. Mellado, and E. Fernandez-Medina, "Security analysis in the migration to Cloud environments," *Future Internet*, 2012.

[6] M. Hajjat, X. Sun, Y. Sung, D. Maltz, S. Rao, K. Sripanidkulchai, and M. Tawarmalani, "Cloudward bound: planning for beneficial migration of enterprise applications to the Cloud," vol. 40, pp. 243–254, ACM, 2010.

[7] A. Juan-Verdejo and H. Baars, "Decision support for partially moving applications to the cloud: the example of business intelligence," in *Proc. of the int. workshop on Hot topics in cloud services*, pp. 35–42, ACM, 2013.

[8] V. X. Tran, H. Tsuji, and R. Masuda, "A new QoS ontology and its QoS-based ranking algorithm for web services," *Simulation Modelling Prac. and The.*, 2009.

[9] M. Zeleny and J. L. Cochrane, *Multiple criteria decision making*, vol. 25. McGraw-Hill New York, 1982.

[10] CSMIC, "Introducing the service measurement index." http://www.cloudcommons.com/web/cc/about-smi, 2011. Online; Last Accessed 11-December-2013.

[11] A. Li, X. Yang, S. Kandula, and M. Zhang, "Cloudcmp: comparing public cloud providers," in *Proc. of the 10th conf. on Internet measurement*, ACM, 2010.

[12] M. Menzel and R. Ranjan, "Cloudgenius: Automated decision support for migrating multi-component enterprise applications to Clouds," *Technical Report*, 2011.

[13] T. Binz, F. Leymann, and D. Schumm, "Cmotion: A framework for migration of applications into and between Clouds," pp. 1–4, IEEE, 2011.

[14] M. Klems, J. Nimis, and S. Tai, "Do Clouds compute? A framework for estimating the value of Cloud Computing," *Designing E-Business Sys.*, 2009.

[15] T. Saaty and L. L. G. Vargas, *Models, methods, concepts, and applications of the AHP*. Springer, 2001.

[16] J. D. Mooney, "Bringing portability to the software process," *Dept. of Statistics and Comp. Sci.*, 1997.

[17] E. Wittern, J. Kuhlenkamp, and M. Menzel, "Cloud service selection based on variability modeling," in *Service-Oriented Computing*, pp. 127–141, Springer, 2012.