Cloud Infrastructure for Providing Tools as a Service: Quality Attributes and Potential Solutions

Muhammad Aufeef Chauhan
Software and Systems Group
IT University of Copenhagen
Copenhagen, Denmark
+45 4223 3348
muac@itu.dk

Muhammad Ali Babar
Software and Systems Group
IT University of Copenhagen
Copenhagen, Denmark
+45 7218 5107
malibaba@itu.dk

ABSTRACT
Cloud computing is being increasingly adopted in various domains for providing on-demand infrastructure and Software as a service (SaaS) by leveraging the utility computing model and virtualization technologies. One of the domains, where cloud computing is expected to gain huge traction is Global Software Development (GSD) that has emerged as a popular software development model. Despite several promised benefits, GSD is characterized by not only technical issues but also the complexities associated with its processes. One of the key challenges of GSD is to provide appropriate tools more efficiently and cost-effectively. Moreover, variations in tools available/used by different GSD team members can also pose challenges. We assert that providing Tools as a Service (TaaS) to GSD teams through a cloud-based infrastructure can be a promising solution to address the tools related challenges in GSD projects. Through an extensive review of the relevant literature on GSD and Cloud Computing, we have identified a set of quality attributes and potential architectural solutions for a cloud-based infrastructure that can provide TaaS to GSD teams. This paper outlines the promised benefits of TaaS to GSD teams, describes the expected quality attributes of a cloud-based infrastructure and how these quality attributes can be achieved. We also present a reference architecture for a cloud-based infrastructure to provide TaaS in GSD projects and describe its use with a detailed scenario of GSD projects.

Categories and Subject Descriptors

General Terms
Design, Standardization.

Keywords
Cloud Computing, Global Software Development (GSD), Software Engineering (SE), Software as a Service (SaaS), Tools as a Service (TaaS).

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
Copyright 2012 ACM 978-1-4503-1568-5/12/08 ...$15.00.

1. INTRODUCTION
Cloud computing promises to enable organizations to have on-demand scalable IT (i.e., computing, network, and storage) resources without making huge upfront investment [19, 26, 34]. Like many other domains that stand to greatly benefit from cloud computing in several ways, it has been stated that cloud computing can also help organizations to address several processes and technical challenges associated with Global Software Development paradigm [22]; which has become an increasingly popular model of developing software with geographically distributed teams. GSD is characterized by geographical, temporal, and cultural distances that usually result in several kinds of challenges related to collaboration, communication, coordination, and management of processes [11, 17, 22, 31]. One of the key challenges of GSD is to provide appropriate tools to support the effective and efficient working arrangements for GSD teams [32]. For carrying out software development activities by following specific organizational processes, organizations require specific set of tools. When organizations with different processes, using different set of tools decide to collaborate with each other in the context of GSD; they have to align their processes and agree on standardized set of tools. Whilst organizations working on big projects with sufficient resources may be able to afford the required changes, small organizations are unlikely to be able to adjust their tool support for each project. Hence, the organizations can be in a better position to align their tools if they are able to acquire Software Engineering (SE) and other supportive tools whenever they need them without upfront investment. We argue that cloud computing related technologies can be exploited to provide Tools as a Service (TaaS) in GSD projects.

The benefits of cloud computing to address GSD specific challenges have been reported in the literature [22, 27]. Hashmi et al. describe GSD as a product as well as a service in cloud [22]. According to them, geographical and temporal issues are one of the major GSD challenges. Geographical distances result in communication gap, ambiguity in domain and product requirements, and challenges in transferring domain knowledge. Temporal issues result in less visibility of product artifacts, risk of poor visibility and reduced product quality. The authors also discuss the utilization of cloud-based infrastructure for achieving business and technology alignment, interoperability and diversification of tools [22].

The use of tools in GSD has been described along with different lifecycle phases of software development in [27] where the tools have been clustered into four groups. Each group is aimed at addressing specific GSD challenges. An effort to optimize software development tools requires an integrated set of tools. The tools optimization effort is addressing the technical issues related
to software development and highlights the need for SE tools to provide solutions to one specific aspect of software development, for example, testing tools or requirements management tools. The project optimization effort includes the tools that are needed for project management and governance. Optimization of GSD highlights the need of Software as a Service (SaaS) approach to solving GSD issues. We assert that the availability of TaaS can allow organizations and individual engineers to have easy alignment of processes and tools because there would not be any need to setup new tools and infrastructure.

We have stated that there has been an initial attempt to identify the areas and approaches where cloud computing can be utilized to support GSD [22]. However, there has not been any attempt, to the best of our knowledge, to systematically analyze the relevant literature on cloud computing and GSD for identifying the kinds of quality attributes (i.e., extra-functional properties) and potential architectural solutions for building a cloud-based infrastructure for providing TaaS to GSD. In order to identify a set of quality attributes and the potential architectural solutions for enabling organizations to materialize the promised benefits of cloud computing for GSD teams, we have extensively reviewed the literature on applications and platforms development for hosting tools as service in clouds [7]. Based on the literature review, we have determined a set of key quality attributes of the cloud infrastructure that is capable of supporting TaaS for GSD teams. To provide guidelines on how to achieve the identified quality attributes while designing a cloud-based TaaS infrastructure, we have identified a set of architecture constructs, and, where needed, pointed out and discussed the relevant technologies. Whilst TaaS are also expected to benefit collocated software development teams, e.g., Model-Driven Engineering (MDE) tools, the focus of our research reported in this paper is on GSD teams.

Hence, the main goal of this paper is to report and discuss the quality attribute of a cloud-based infrastructure for providing TaaS in GSD environments and a set of potential architectural strategies for designing an appropriate architecture. We assert that we have identified a set of “base-line” quality attributes from the literature which can be assumed to report the expectations and/or opinions of several experts working in the area. We expect these quality attributes to evolve overtime based on the community’s feedback and agreement among experts, researchers and practitioners. The identified quality attributes and potential solutions are expected to make contributions to the research and practice for leveraging cloud computing in GSD in several ways. For example, the set of identified quality attributes can be used as a starting point for identifying the relevant quality and functional requirements for a cloud-based TaaS infrastructure and evaluating the designed architectural solutions. The quality attributes and architectural solutions reported in this paper can also stimulate new research directions for exploring new and novel ways of aligning GSD processes and tools by leveraging cloud computing technologies.

Organization of the paper is as follows. Section 2 provides a brief description of Global Software Development (GSD) and Section 3 provides an overview of the advantages that can be achieved by hosting software development tools in clouds. Section 4 outlines a set of quality attributes of the cloud infrastructure hosting the tools and gives an overview of the potential solutions for achieving the identified quality attributes. Section 5 presents a reference architecture and its analysis with respect of GSD scenarios. The last section concludes the paper and highlights a few aspects in which TaaS is different to SaaS.

2. GLOBAL SOFTWARE DEVELOPMENT

Global Software Development has emerged as one of the most popular model of developing software intensive systems. All types and sizes of organizations are adopting a geographically distributed software development format [23]. Apart from the desire to achieve the promised benefits such as extending working hours and/or teams with the required set of skills, taking advantage of low cost destinations, and being closer to a significant market, GSD is also gaining momentum as a result of mergers, acquisitions, alliances, and several kinds of market demands [3-4, 18].

GSD projects are often large-scale, and global development leads to significantly increased complexity for project teams, who may have to face several kinds of new challenges characterized by the need of new and novel coordination, collaboration, and communication mechanisms to support different software development activities undertaken by team members located at multiple sites, each with its own time zone and with different cultural backgrounds [5, 15, 24]. The distances (such as temporal, geographical, and cultural) commonly found in a GSD project usually result in challenges (such as increased inter-dependency, absence of face-to-face meetings, and linguistic competency) that may not be encountered by the teams developing software in more traditional way, e.g., in a co-located arrangement. Other kinds of potential challenges of GSD can be delayed feedback, restricted communication, less shared project awareness, difficulty of synchronous communication, inconsistent development and build environments, and lack of trust and confidence between sites.

Hence, it is important to provide adequate and suitable tool support to GSD teams. There has been significant amount of efforts allocated to adopt tools from other disciplines such as Computer Supported Cooperative Work (CSCW) and/or develop tools for supporting GSD teams. A study of reviewing the tools developed and/or used for support GSD teams identified 384 tools that have been used for GSD teams [32]. Since GSD teams may have different organizational structures and working patterns that is likely to have some conflicting aspects including the use of several different tools for the same purpose. Hence, this research is being carried out based on the premise that it would be quite helpful and beneficial if tools required for supporting GSD teams are available as Tools as a Service (TaaS) hosted in a cloud infrastructure.

3. ADVANTAGES OF TaaS

Cloud-Based provision of TaaS can enable software development organizations to acquire tools whenever the tools are required. This section describes some of the potential advantages of providing TaaS in GSD [14, 22].

3.1. No Fixed License Fees

Software development consists of multiple phases and each phase requires some specific set of tools, as the tools used in one phase may not be used much when a project is in another development phase, especially the tools that are used in the later development phases. For example, in traditional waterfall and spiral development process models [28], when a project is in the requirements engineering phase, the tools required in design and development phases are not used. Organizations still need to acquire the licenses for the tools for the whole span of a project. Similarly, when a project is in the testing phase, the requirements engineering and design tools are not used frequently by a large number of stakeholders. Since cloud computing utility model can support the provision and acquisition of tools for a specific time frame and an organization can have the flexibility of acquiring a specific set of tools at the point when it is needed [22], it means the organizations will be paying for the usages of the tools when it is really used. Some development phases may need more resources,
for example, in order to simulate test cases associated with computing and data-intensive use cases, additional hardware and software resources are required. These resources are utilized only for a small fraction of time as compared to overall system development life cycle. Moreover, when multiple sites are involved in GSD, it may not be cost effective to replicate tools as well as associated hardware resources at all location [22]. Using the computing as a utility model, organizations can acquire additional resources during the testing phase for simulating the resource hungry test cases.

Furthermore, organizations involved in GSD require different tools in accordance with the requirement of each specific project. Configuring and maintaining a large variety of tools are resource and time intensive undertakings. When GSD projects are not in active stage, the tools and associated infrastructure still occupy the resources. These constraints may limit an organization’s options to work in different business and application domains. The availability of TaaS through a cloud-based infrastructure can help organizations to acquire and release the desired set of tools according to the specific requirements of projects and only for the phases when the tools are really needed.

3.2. Alignment of Tools with Processes
Organizations involved in GSD have to align their software development processes with each other in order to effectively work on product artifacts [22]. As a consequence, organizations also require some tools for facilitating the processes. Software design tools and software testing tools are common examples of tools that are closely tied with process. Also, for a small sized organization involved in GSD projects, it is not cost effective to have a specific set of tools for each GSD project they are working on. If tools are available as TaaS, they can be acquired according to the process requirement of each single project and can be released afterwards.

3.3. Artifacts Traceability
Organizations involved in GSD projects using different set of tools cannot provide traceability between artifacts produced in different development phases. A set of integrated tools across different sites can ensure traceability within a site as well as across multiple sites.

3.4. Working with Sensitive and Massive Data
Dealing with sensitive data in GSD projects is a challenging task. There are certain laws that restrict data movement outside a particular geographic location. Cloud based infrastructure can help in addressing these challenges. For example, Cloud-based software testing tools can facilitate testing using the data that is of sensitive nature. These tools can deploy system components handling sensitive data in cloud inside the same geographic location where data are present to run test cases and collect the resulted metrics while being accessible from outside that region.

GSD projects dealing with huge data volumes can also take advantage of cloud computing. For example, testing of a specific part of an application may require moving a large amount of data from one geographic location to another. Moving a huge volume of data waste a lot of time and network bandwidth. Cloud-based tools can eliminate the need of moving the data from all sites of a GSD project and allow developing an application inside the region where data can be easily and rapidly accessed.

3.5. Working with Expensive and Technology Sensitive Hardware
Software artifacts developed in a distributed environment may be part of a very complex system requiring integration with expensive and sensitive hardware. Replicating of expensive and technology sensitive hardware at each location may not be possible because of cost or sensitive nature of the hardware technology. If such hardware resources can be connected with software development tools on a private cloud, it can allow remote sites to participate in the development of such a project. Stakeholders in remote areas can perform their activities from remote locations using tools that are hosted in a cloud-based infrastructure.

3.6. Support for Awareness and Collaboration
Making all sites of a GSD project aware of the development activities and the status of the artifacts produced is a challenge [31]. Awareness and collaboration between distributed teams can be enhanced by improving communication, cooperation, and coordination [31]. Studies on GSD have shown that implicit information about project and communication that exist between collocated teams is not available to remote teams [20]. That is why emails, text chats, and instant messages are used as informal ways to increase awareness [20]. However, information that is exchanged through emails, texts and instant messages is detached from the actual artifacts and is not traceable. Tools and editors are designed to provide visual representation of participants’ actions and allow them to view each other’s work [20]. The traditional desktop based tools and editors require extensively exchange information with each other for providing awareness to other software engineering tools because artifacts are scattered at multiple geographic locations. In case of TaaS hosted in a cloud, wherever users want to access artifacts, they will have the most updated view without any additional effort. Tools hosted in a cloud, minimize the burden of data exchange between sites, hence reducing latency delays. Moreover, incorporating additional features of awareness in TaaS can help establish and maintain traceability between artifacts and collaborative exchange of information because they can share underlying data structures provided by a cloud infrastructure hosting TaaS.

3.7. Inter-Organization Knowledge Ecosystem
Organizations involved in software development maintain their in-house knowledge bases for internally sharing knowledge. When organizations are collaborating with each other, access to the knowledgebase across multiple organizations can be very
beneficial. Sharing knowledge from private knowledge base is not trivial because it may require specialized tools (developed by an organization for their in-house use) or require data transformations (knowledge base is maintained in proprietary storage formats). If knowledge acquisition and management tools are hosted in a cloud whose infrastructure has integration support with the tools being used by organizations, knowledge sharing across organizations becomes convenient and transparent to the end users. Cloud infrastructure also makes it easier to manage access to information when the organizations do not work collaboratively.

Figure 1 depicts a high level view of how cloud enabled infrastructure can encompass different tools for supporting software development activities.

4. QUALITY ATTRIBUTES AND POTENTIAL SOLUTIONS

It has been discussed that the provision of TaaS can facilitate GSD. However, achieving this task is not straightforward. There are certain quality attributes that the tools as well as a hosting cloud infrastructure should support. Based on our literature review, we have identified an initial set of quality attributes required for a cloud-based infrastructure for providing TaaS. The term “quality attributes” for this research has slightly more wider scope than the term used in the software architecture discipline. Here we mean the extra-functional properties expected of a TaaS cloud infrastructure. For achieving extra functional properties, a TaaS cloud-based infrastructure needs to implement specific architectural constructs. The sub-sections of this section provide a brief overview of the quality attributes required of and the potential solutions to designing an infrastructure capable of supporting TaaS.

4.1. Supporting Multiple Organizations

Tools hosted in clouds are required to support multiple organizations as well as multiple users inside one organization. In a cloud computing environment, this concept is referred as Multi-tenancy. Multi-tenancy ensures isolation between services and data being used by these services [1]. Multi-tenancy is important for all types of software engineering tools hosted in clouds. However, it is more critical in case of integrated development environments (IDEs) and testing tools. Other than working on a set of artifacts, development and testing tools also generate executable entities (executable files), which can hinder the operations of other tools hosting a cloud as well as executable entities generated by other tools if they not properly insulated from the services and artifacts of other tenants.

Safety of the data being used by tools in a cloud and the artifacts that are produced are also important. Each organization can have its own specific requirements related to the safety and storage of the data. For certain types of projects, it would be sufficient to provide data isolation whereas some other projects would require to store sensitive data in private clouds.

Potential Solutions:

A simple way to achieve multi-tenancy in a cloud is to assign separate service instance of tools to each tenant [1, 29]. Users belonging to a tenant use the same instantiated tool and corresponding services. Information of services corresponding to tenant is maintained in a service registry. A security handling service can be implemented to ensure secure access and to validate access rights of users belonging to one particular tenant. Each project can also have its own specific requirements for persistence that can be abstracted by database driver services responsible for defining mapping between service instances belonging to each tenant and corresponding persistence units [12]. Database driver provides abstraction of isolated implantation of services and separate persistence functionality from the business logic of components.

4.2. Maintaining Different Versions of the Tools

There is a large variety of tools used in software development in general and GSD in particular (e.g. collaboration, coordination, and communication tools as well as general SE tools). Each of these tools can have different versions and different features bundles according to the requirements of different stakeholders/tenants. Each of them can request different tools as well as different versions of the same tool. So, in order to have a comprehensive support for hosting these tools, the hosting cloud environment should be able to maintain different tools, their versions and feature bundles of the tools. It is evident that maintaining this kind of complex environment in a cloud increases complexity of the hosting infrastructure.

Potential Solutions:

The version management services capable of maintaining partitioning between different versions of a tool as well as bundle specific version of the multiple tools are needed to support tools versioning [2]. These services are also needed to manage traceability between tools used in different development phases as subscribed by GSD tenants. As there can be multiple featured packages of the same product version; hence, versioning services also need to take care of feature bundling of products according to the requirements of a project.

4.3. Combining Multiple Tools

In order to provide an end-to-end solution covering all the phases of software development life cycle, the tools hosted in clouds dealing with one phase should be able to integrate with tools serving preceding and proceeding development life cycle phases. As each activity of a software development project uses the artifacts produced by the preceding activity; there is a need to provide traceability between the artifacts used in different activities of the software development lifecycle. There is also a need to provide compatibility between tools that can be used in same development activity. This will enable development teams to opt tools of their own choice, which are more suitable to projects requirements and skill set of the stakeholders.

Potential Solutions:

Underlying data structures of the tools hosted in a cloud are required to be made compatible to each other for providing traceability and compatibility. Having compatible data structures makes it easier to bundle these tools together to provide a complete solution for one project belonging to a particular tenant. However, in certain cases when GSD organizations are using some in house or third party tools to carry out development activities of one particular phase, the cloud-based TaaS infrastructure should be able to provide hooks for supporting traceability between different design artifacts. These hook points will enable writing glue code services that can make conversions when data is exchanged. As a result these glue code services would link artifacts (e.g. design documents) with other artifacts of the preceding and proceeding phases.

4.4. Compatibility with Commercial Tools

There are a large number of commercial and open source tools available that can support software development activities. For facilitating organizations to use hosting infrastructure for TaaS, the infrastructure should support integration with tools suite of different organizations. This integration is important for following reasons. 1) For smooth functioning of organizations, it is important...
that they do not have to change the existing SE tools for every new project. If the infrastructure supports integration with existing tools, the tools can be used as an interface to the infrastructure while underlying data structures are maintained in clouds. It makes it easier to have access to shared artifacts across multiple development sites using different tools. 2) It may not be possible to have alternate of every single tool available in clouds as TaaS. Integration support for existing tools with infrastructure makes infrastructure adoption easier and smooth.

Potential Solutions:
In order to provide compatibility with commercial tools, a hosting infrastructure should expose its features using platform neutral APIs. These tools can use infrastructure features to share artifacts and can integrate with other tools hosted in a cloud-based infrastructure by using the provided APIs; e.g. by implementing plugins.

4.5. Enabling Tools to Work With Private Data
The certain types of applications may require product artifacts stored inside one or more development sites because of sensitive nature of the applications being developed or because of organizations internal security policies. In such cases, a cloud-based TaaS infrastructure should be able to integrate with private storage cloud (or simple storage methods) of the organizations.

Potential Solutions:
For achieving decentralized arrangement of data and/or to protect sensitive data, processing services inside each tool can be implemented as a workflow based system. Services processing sensitive data can be deployed inside the premises of a development site or client organization, and non-secure data can reside in a private cloud. It requires a client side engine to manage tools configuration on client side according to data sensitivity requirements [21]. The services inside client side engine expose only unsecure information to the external tools and services. For further security enhancement, the services hosted inside organizational premises should interact only with immediate upstream and downstream services of tools hosted in a cloud-based TaaS infrastructure [13]. Data management services in a cloud infrastructure handle integration of cloud side data with privately hosted data in a transparent fashion and support smooth functioning of the tools.

4.6. Support for Multiple Types of Persistence Methods Accessible Through Tools
Software development tools, especially Integrated Development Environments (IDEs), hosted in a cloud environment may need to have access to a wide variety of persistence schemes. For example, they need to access multiple types of relational or object-relational databases. There should be mechanisms to make these tools easily interoperable with several kinds of databases as each application being developed may have its own persistence requirements; e.g. an application may requires MySQL database for implementing the required solutions.

Potential Solutions:
Infrastructure as a Service (IaaS) providers like Amazon [14] are offering different relational database including Oracle and Microsoft SQL server hosted in cloud-based infrastructure. Tools hosted in a cloud-based infrastructure can be integrated with commercial database services. In order to provide an isolation for abstracting multi-tenancy related implementations, multitenant database drivers can be added for each database [12]. The intermediate layer takes care of providing isolation between data belonging to different tenants.

4.7. Access by Multiple Types of Devices
Software development teams may need to use multiple devices to access different artifacts. For example, different types of hand-held and mobile devices like tablets and smart phones, which can support different activities of software development projects. The use of mobile devices during requirements engineering activities can be very convenient for requirements elicitation. An easy access to tools though mobile devices enables stakeholders to directly input requirements into a requirements management tool rather than using intermediate documentation. It can help to identify conflicting requirements during requirements gathering processes and provide a better opportunity to engage stakeholders in negotiation. Hence, a cloud infrastructure capable of hosting requirements engineering and management tools should be able to configure its services’ configurations for supporting multiple devices.

Potential Solutions:
For tools to be accessible through multiple types of devices, an infrastructure hosting the tools should be able to dynamically distribute processing load among client devices and on cloud-based infrastructure [30, 33]. When SE tools are being accessed from powerful client devices in terms of processing and memory resources, their resources can be utilized to minimize the utilization of cloud resources. For having optimal cloud configuration, a cloud-based infrastructure should be able to dynamically configure tools hosted in cloud so that more processing load can be shifted onto end user devices with the help of service management components. An infrastructure should also provide hooks for implementing algorithms determining optimal configuration strategy of components in a cloud-based infrastructure [16]. When multiple devices are accessing a system, there is also a need for response transformation services that can transform responses in the format recognizable by client devices.

4.8. Service Level Agreement Compliance
Tools hosted in a cloud can be used by multiple organizations and by more than one project inside an organization. Each client organization can have its own quality of service (QoS) requirements for tools. Often, these requirements are not static; rather, they vary from project to project and also depend upon phase of the software development lifecycle of a single project. As a result, each of the tools to be provided as TaaS may have its own unique requirements for QoS based on the nature of a project. For example, one project may require a requirements tool to keep the data highly protected and secure. Another project seek throughput as a primary QoS requirement from the same tool. A cloud-based TaaS hosting infrastructure should be able to meet the challenges of such dynamic system behavior and should be able to negotiate and agree on a Service Level Agreement (SLA) for each of the provided tools on the available terms and conditions. The infrastructure should also ensure that the agreed SLA is fully complied with the requirements of each project belonging to a particular tenant and its different phases.

Potential Solutions:
Efficiency, reliability, throughput, number of active users, number of concurrent users, adaptability and integration of tools with other tools as well as services are example of the extra-functional requirements important for a cloud-enabled TaaS hosting infrastructure [9]. Different architecture styles and patterns can be used for achieving these requirements. For incorporating dynamic changes in SLA and QoS requirements, these need to be specified in a machine readable format and a cloud-based infrastructure should be able to reflect the specifications into system dynamically [6]. Scalability of tools and elasticity of the resources can be
achieved by defining scalability rules and elasticity algorithms [9].
These can be taken as input by life cycle management services.
These services deploy actual instances of tools on virtual execution
environments of cloud according to specified requirements.

Tools and services hosted in clouds need to be continuously
monitored for their SLA compliance and require an additional
layer of monitoring services [8, 35]. These services monitor
executions of services according to performance indicators
specified in SLA specifications. Variability of number of tenants
accessing the GSD TaaS applications makes these services
significantly important. If an inconsistency is detected during
services execution, the monitoring services can redefine elasticity
rules and as a result, life cycle management services can adjust
tools deployments and associated components in a cloud. In some
cases, TaaS in clouds may need to use infrastructure resources
from public or private clouds external to a system (for example,
databases provided by public cloud providers) and failure of any of
the external services can affect availability and reliability of
internal services. For handling abnormal situations when services
fail to comply with their SLA, a service management engine is
required to keep services aligned to their QoS requirements by
provisioning required resources and in worst cases, it may need to
find alternative services. The concept of accountable third party
services by assigning rewards and penalties to them based on their
performance can improve service selection procedure [35].

4.9. Specifying and Modeling SLA

In order to support automated service brokerage, appropriate SLA
specifications should be defined in a machine-readable format.

Potential Solutions:
The semi-structured and formal approaches for SLA specifications
are described in [6]. Three key requirements for automated
resource provisioning are the ability to specify the structure of
meta-data, specification of resource requirements, and
specification of constraints between abstract syntax and domain
elements used to model operations of cloud infrastructure
components. Open Virtualization Format (OVF) is used to describe
resource requirements of applications using an XML based simple
structure. Use of architecture description language (ADL) is a
more formal way of specifying syntax and elasticity rules in a form
of key performance indicators (KPIs). A cloud-based infrastructure
for hosting GSD tools should have services and interfaces that can
be used for specifying QoS requirements of individual tools along
with the specifications of tools bundling according to specific
requirements of tenants working on a GSD project.

4.10. Tools Bundling

Software tools hosted in cloud infrastructure should have the
ability to be offered to end users as a group of tools to provide a
comprehensive solution to the organizations.

Potential Solutions:
Virtual machine images can be used to facilitate users when
multiple tools are used together [14]. The virtualized images
containing Graphical User Interfaces (GUIs) and light-weight
client components can be configured according the requirements of

Figure 2: Infrastructure for Handling Virtualized Tools and Services
tools of a project as well as according to the requirements of one particular phase of a project. The bundling capability frees users from the hassle of configuring clients of tools hosted in a cloud. It also makes it easy for the stakeholders to switch between projects when they are working on multiple projects in parallel.

Table 1 provides a summary of the above discussed quality attributes and the potential solutions for achieving those quality attributes. Based on the identified solutions, a representation of the overall system architecture is presented in Figure 2.

### Table 1: Potential Solutions to the Required Quality Attributes

<table>
<thead>
<tr>
<th>Quality Attributes</th>
<th>Potential Solutions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Multi-tenancy</td>
<td>Isolated tenant-specific instances of tools and multi-tenant database drivers.</td>
</tr>
<tr>
<td>Versioning</td>
<td>Version specific features bundling.</td>
</tr>
<tr>
<td>Compatibility with</td>
<td>Platform neutral APIs.</td>
</tr>
<tr>
<td>commercially available</td>
<td></td>
</tr>
<tr>
<td>tools</td>
<td>Tools consist of composable services that can be deployed on private clouds.</td>
</tr>
<tr>
<td>Working on decentralized</td>
<td>Integration with database services offered by public cloud providers.</td>
</tr>
<tr>
<td>artifacts</td>
<td></td>
</tr>
<tr>
<td>Supporting multiple</td>
<td>Composable and adaptable cloud based services and configurable GUIs.</td>
</tr>
<tr>
<td>databases</td>
<td></td>
</tr>
<tr>
<td>Supporting multiple devices</td>
<td></td>
</tr>
<tr>
<td>Service Level Agreement</td>
<td>Autonomous composition of tools and services in cloud according to SLA specifications.</td>
</tr>
<tr>
<td>(SLA) Compliance</td>
<td></td>
</tr>
</tbody>
</table>

### 5. INFRASTRUCTURE ANALYSIS

In this section, we explain the uses and the potential advantages of using a cloud-based infrastructure for a GSD with the help of a scenario related to an organization X, which is located in China and is working on two GSD projects. Project A purports to develop a web-based application for a public sector organization in Denmark. This project is being developed using open source technologies and requires access to private data and other system services during unit and integration testing in different releases. Organization X is also working as an industrial partner with a Danish company on a project B that is related to the streaming of real-time information of traffic on highways and roads in metropolitan areas. This project has two motivations: 1) make traffic information available to commuters through web-based interfaces so that they can follow routes with less congestion during busy hours of a day; 2) control traffic signals in metropolitan areas using traffic information.

Both projects bring specific challenges for organization X. Project A requires access to sensitive data and other sensitive services being used. Danish data security and privacy laws restrict the transfer of such data outside of Denmark. If the data and services cannot be made available to organization X, complete development cannot be carried out on their premises in China and requires development of artifacts dealing with sensitive data and services inside Denmark. Project B requires access to the traffic data as well as access to traffic signal control network. If development to be carried out in China using the traditional development approaches, it requires simulation of real traffic network and at a certain point a team of testers would be needed in Denmark for integration and system testing. Organization X in China may also need to shuffle developers within projects depending on the workload and criticality of the release. Configuration of development environments requires significant effort during resource shuffling. A cloud-based infrastructure shown in Figure 2 can address these challenges. The infrastructure can be hosted in Denmark to link it with storage cloud, with other public sector services, traffic monitoring sensors and traffic control signal network. Middleware infrastructure layer of the system architecture shown in Figure 2 is the core layer of the architecture responsible for offering virtualized bundled tools to end-users.

As previously stated, the middleware can be used to abstract the internal management of the tools and services from the end user. Each project is treated as a tenant. Tools and other requirements of the projects can be specified in SLA. Other than offering TaaS, this infrastructure can also support integration with external services, expensive and technology sensitive hardware resources, and private storage clouds.

In Figure 2, Host Type 1 corresponds to the virtual node for the developers working on the project B that is being developed using proprietary development tools. Other than managing virtualized tools as services, the infrastructure also provides connection to traffic monitoring sensors and a grid of traffic control signals. Given the availability of this infrastructure, developers at remote sites can have access to same resources as the onsite developers without replicating expensive sensitive/hardware and simulations for external system services that are sensitive in nature. Host Type 3, represents computer tablets being used by project managers to view software artifacts.

### 6. CONCLUSIONS

Global Software Development has become extremely popular area of research and practice. GSD mode of developing software has several inherent challenges as a result of geographical, cultural and temporal issues [10]. An appropriate tool support can play an important role in addressing some of the GSD challenges [25, 32]. However, organizations may not be able to adjust their tools according to the formulation of GSD teams, which may span inter-organizational members and processes. It has been argued that cloud computing can have quite a number of ways of supporting GSD [22]. We assert that TaaS hosted in a Cloud-based infrastructure can help organizations to provide appropriate tools and get them aligned with the processes without making huge investment. Like any other system development effort, building the proposed infrastructure would need a set of quality attributes. One way of building an initial set of quality attributes expected of a system of general interest is to explore, gather, and synthesize the expectations and opinions of experts in the area as reported in the literature. We have applied this research approach for the research reported in this paper.

This paper has reported a set of “base-line” quality attributes based on an extensive review of the literature on Cloud computing [7].
We have also identified a set of potential solutions for designing appropriate architectural support that can be expected to help achieving the identified quality attributes. By leveraging the knowledge of the quality attributes and the architecture level solutions reported in this piece of work, we have proposed system architecture for a cloud-based infrastructure that can provide TaaS to GSD teams. We have also illustrated the use of presented infrastructure with the help of an illustrated scenario to demonstrate how such an infrastructure can help to achieve some of the identified quality attributes. This paper presents our initial work to support the idea of TaaS in the context of GSD. We expect both the set of quality attributes and potential solutions to evolve over time.

A cursory view of the quality attributes for GSD TaaS may make them appear alike attributes of any other SaaS. However, these are unique in following aspects and pose special challenges.

- TaaS contain executable artifacts (e.g. code) that are continuously being modified by stakeholders and have the potential to breach security mechanisms of the hosting cloud infrastructure. It highlights a need to invent new security protocols, which can allow restriction free application development along with addressing the security of the infrastructure and protecting artifacts of each tenant from unauthorized access.

- TaaS work as a part of tools suit and require integration with other tools in order to carry out development activities smoothly. For example, to develop a very simple web application requires integration of an application server, a database and a version control system with the IDE. Providing this kind of integration without having significant impact on performance is a challenge and it becomes more complex with the increase in number of tools and availability of multiple versions of each tool.

- TaaS allows development of artifacts that in turn can consume resources during their development and testing. For example, executing a piece of code being developed by a TaaS IDE or database schema generation by an entity relationship (ER) diagram modeling TaaS. This resource consumption is different than resources consumption by TaaS itself and it is hard to predict their resource requirements. It raises the need of new pricing mechanisms capable of distinguishing between resource consumption by TaaS and resource consumption by the artifacts that are generated as a result of using these tools.

- SLA specifications for TaaS need to be addressed differently than SaaS. There are two main reasons for this distinction. First, TaaS produce artifacts that need to be executed during their development and may seek unexpectedly large number of additional resources. It needs separate SLA specifications for the tools that users will subscribe to and for artifacts generated as a result of using the tools. Secondly, as tools for software development are often used together, so SLA should be changed dynamically according to bundling of tools and as a result of changes in the bundling. Hence, there is a need to investigate how dynamic and changeable SLA specifications can be applicable to cloud-based environment, to protect interests of both TaaS providers and TaaS consumers.

- Integration with other tools is one of the major differences of TaaS from SaaS. SaaS provide integration with homogeneous products that are often developed and maintained by same organizations. Whereas, envisioned TaaS for software development requires integration with tools provided by a large number of proprietary and open source solutions providers.

We assert that this work makes at least two important contributions to the efforts aimed at leveraging Cloud computing in GSD: 1) it has identified an initial set of general quality attributes for a cloud-based infrastructure that can provide TaaS. The identified quality attributes can provide a starting point for determining the quality attributes for a specific cloud-based TaaS infrastructure; 2) the reported quality attributes and the potential solutions are expected to motivate new research directions for exploiting cloud computing to support GSD teams to address their evolving needs of tool support. We believe that the availability of TaaS through a cloud-based infrastructure will also help to improve the productivity of the developers in order to enable organizations to benefit more from the GSD paradigm. We also foresee the utilization of the proposed infrastructure by organizations involved in heavy industry automation as it provides flexibility to involve globally distributed teams in the development process without setting up sophisticated distributed sites.

7. REFERENCES


