BlueShift
Automated Application Transformation to Cloud Native Architectures

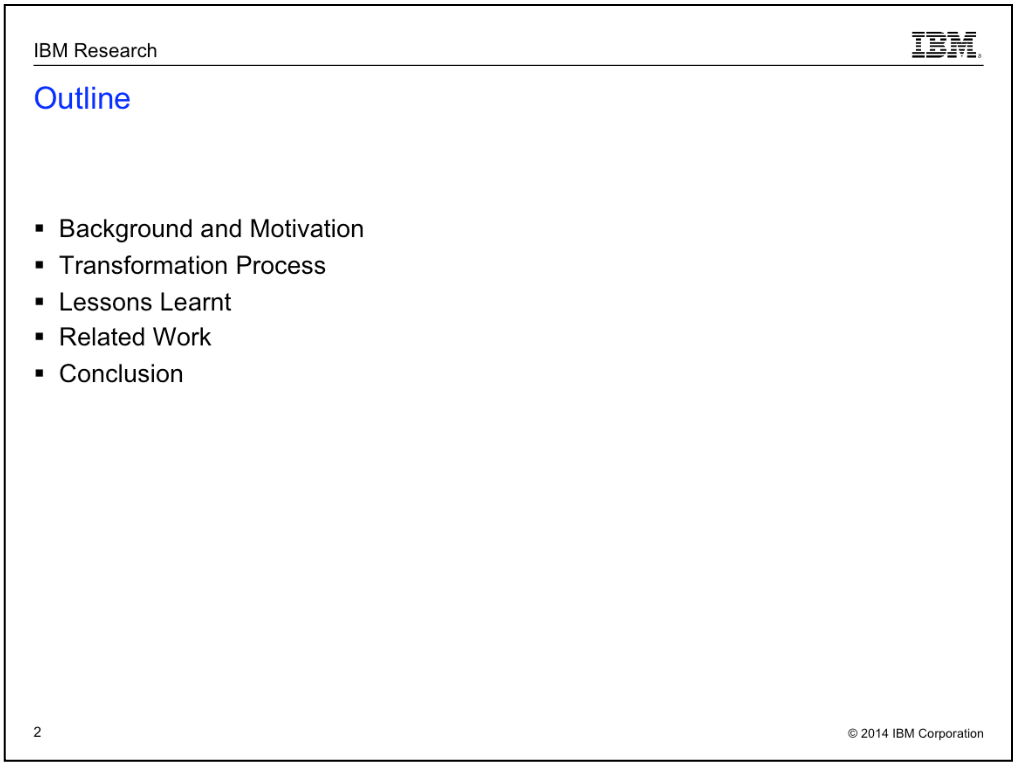
Maja Vukovic, Jinho Hwang, John Rofrano, Nikos Anerousis
{maja, jinho, rofrano, nikos}@us.ibm.com

IBM Research
Yorktown Heights, NY

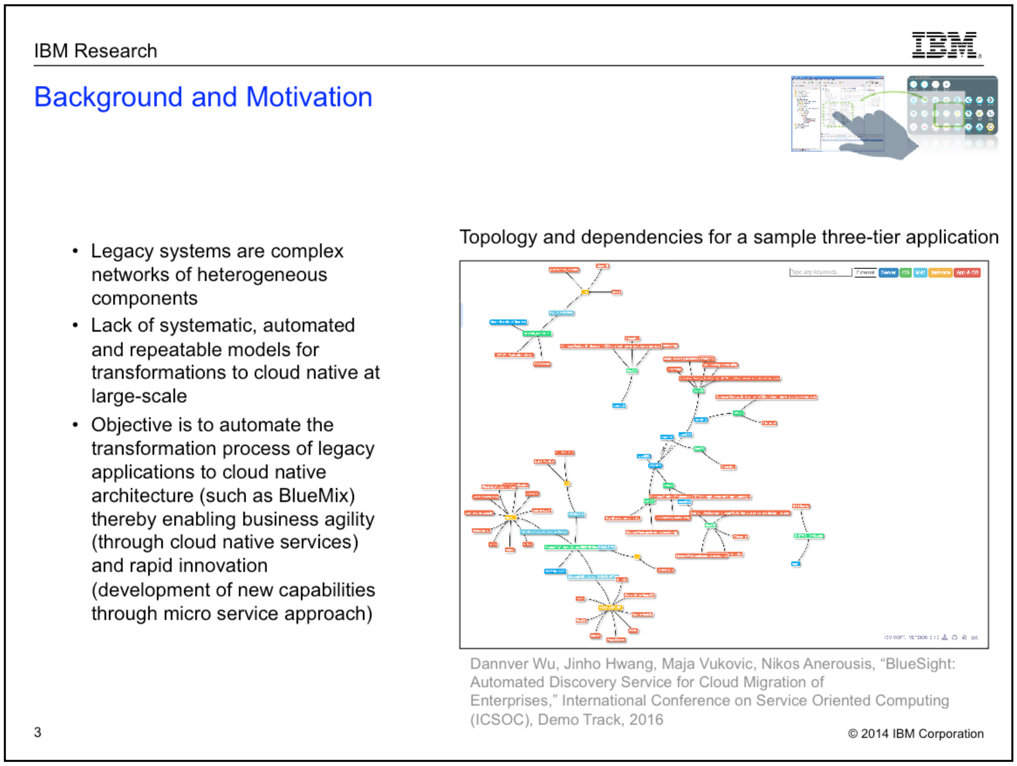
© 2016 IBM Corporation

BlueShift is a service for automated application transformation to a Cloud Native architecture, specifically BlueMix, IBM's cloud platform as a service (PaaS) based on Cloud Foundry open technology.

Most transformation projects rely on highly skilled experts, require manual work and are time consuming. They stop after the migration of the deployment artifacts without leveraging the additional features of the cloud. BlueShift automates an end-to-end transformation process including, application discovery, analysis, artifact transformation and enablement of cloud value-add services. We show how BlueShift automates transformation of Plants by WebSphere application to Liberty Profile Runtime. We further show setup of SessionCache service, and enablement of an automated pipeline and monitoring through cloud native services for the transformed application. We discuss lessons learnt and challenges encountered during this transformation execution.



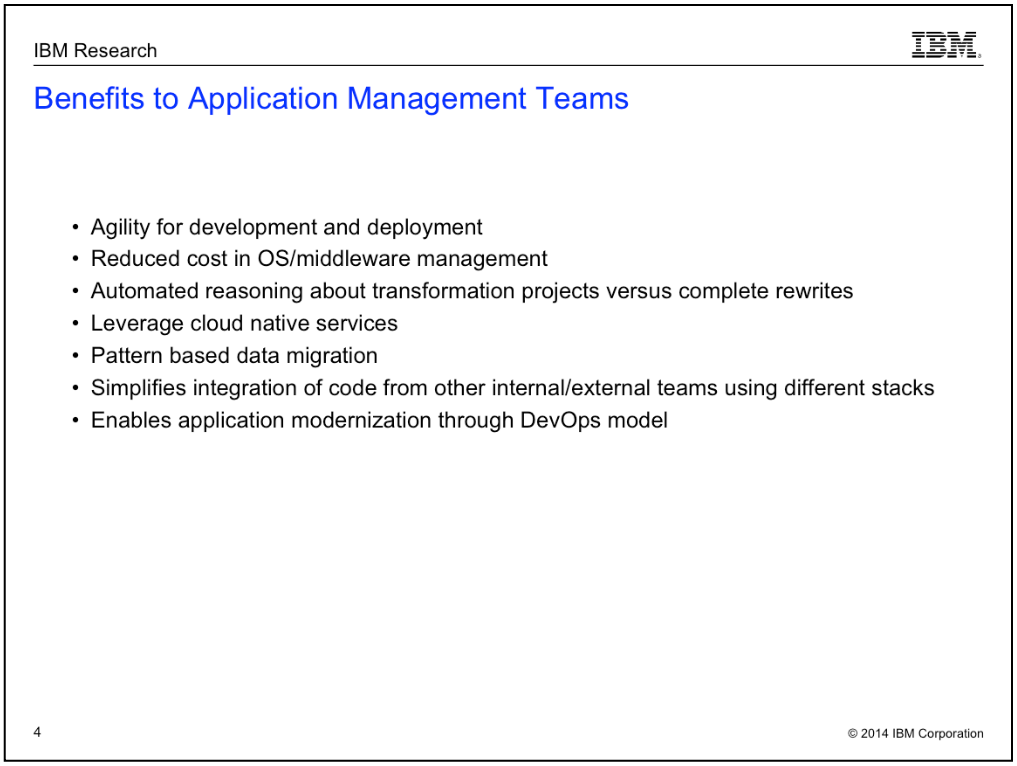
We discuss challenges in large scale transformation projects, benefits of cloud native architectures and our approach. Using a sample WebSphere application PlantsByWebsphere we demonstrate end-to-end process of transformation to Liberty Runtime, enablement of cloud native services and automated pipeline in an automated manner. We discuss lessons learnt in this use case.



An ever increasing number of organizations are choosing to move their workload to the cloud. While the cost of migration is often higher than the capital savings achieved, many organizations select this path to achieve gains in several other dimensions, such as agility (the ability to use DevOps processes and associated tooling to deliver frequent updates their applications – in some cases several times a day), and security (properly architected applications in the cloud can be more secure due to the rigorous compliance testing of the entire stack).

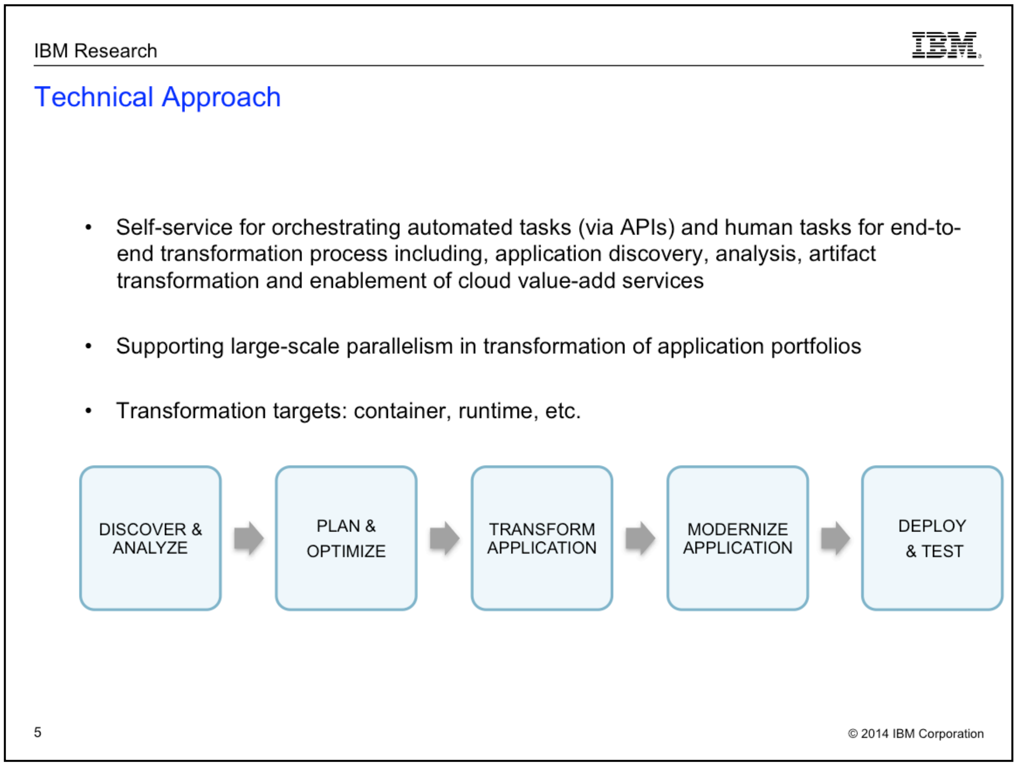
As cloud offerings have evolved, so have the options for moving workload to the cloud. In earlier infrastructure-as-a-service offerings, workload migrations involved either the relocation of virtual images, or rebuilding servers in the cloud (installing middleware and porting the application). More recent cloud offerings that combine DevOps tooling with lightweight runtimes (e.g. containers) offer more opportunities for refactoring or rearchitecting the workload to take advantage of the cloud platform, including a micro-service architecture, the immutability of runtimes and auto-scaling.

Refactoring and rearchitecting workload for cloud requires new capabilities to discover and introspect existing workload. Our work in this paper covers 2 topics: First, the ability to discover automatically the dependencies and versions of infrastructure components supporting applications. Second, the ability to analyze the structure and code artifacts of the workload, and select the optimum transformation path to the cloud.



Our studies demonstrated that an attempt to analyze and rearchitect an existing application for the cloud is a painfully manual and time-consuming process. The development teams that undertake such a task encounter a number of difficulties, from collecting knowledge about the application (which is is a lengthy process when the original development team is no longer available) to understanding which components to keep or refactor in the cloud, to developing and executing a transformation plan. The process is repeated from the beginning for every new workload under consideration.

Our work offers a highly automated alternative to the manual transformation process. It begins with the ability to do deep discovery of a workload, including its internal code structure and dependencies. Once the internal structure has been identified, it is easier to create a transformation plan of how individual components can be modified and make use of native cloud services. Finally, the plan can be executed by invoking the appropriate transformation patters. All in all, the team responsible for the transformation will benefit for greatly reduced times throughout the process, and the ability to parallelize these tasks.

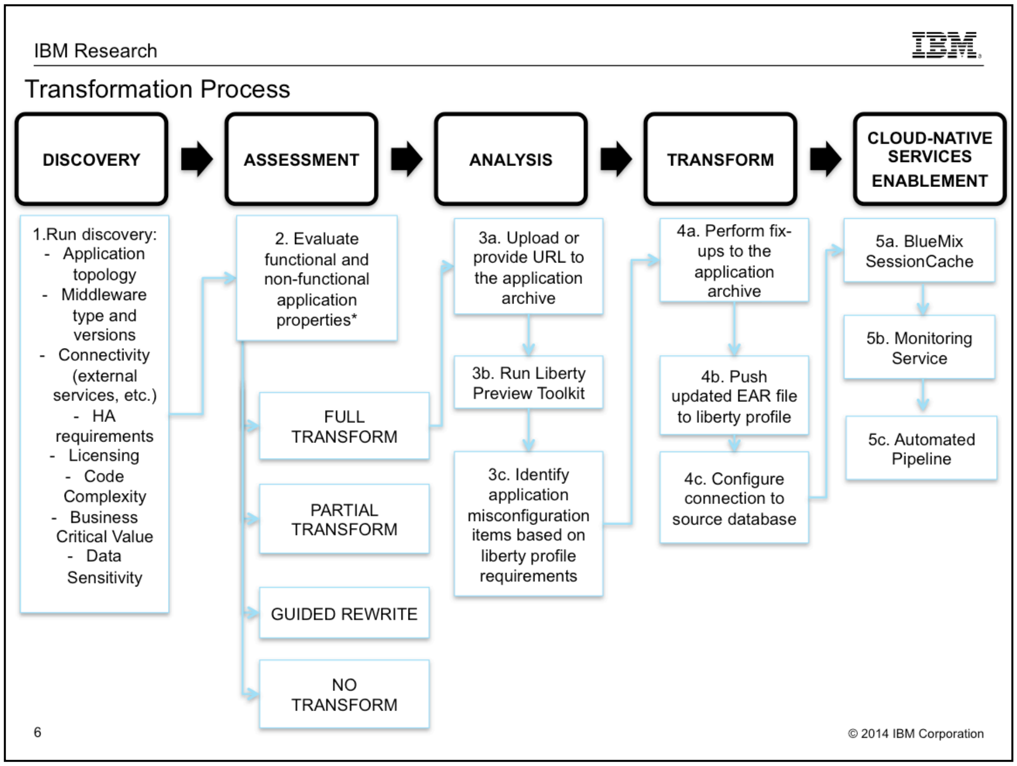


BlueShift is designed as a self-service capability that orchestrates automated and human tasks to execute analysis of existing applications and their dependencies, identifying applications that impact the business agility, artifact transformation (e.g. code analysis, component extraction, etc.), application modernization and testing. The motivation behind BlueShift is to enable large-scale transformation of application portofolios, but capturing and standardizing key transformation tasks and being able to adapt the transformation process dynamically based on the application profile.

Automated tasks are integrated through REST Application Programming Interfaces (APIs) and enable functionality such as manipulating BlueMix platform, creating a space, application, dependent services and pushing the updated application archive. Manual tasks may involve upload of the discovery data (if not automated), selection of the target environment such as container or a runtime, manual fixups to the application, etc.

Before a transformation process starts, BlueShift collects the discovery data about the application. It captures both horizontal dependencies (server-to-server connectivity) and vertical dependencies (within an application archive component) and reasons about it's suitability to target environment. Advanced analytics are applied to perform migration assessment and reasoning about target matching, given a number of functional and non-functional requirements (e.g. connectivity back to customer's site, complexity of the application, topology, etc.).

There are different modes of transformation – some applications may not be suitable at all for porting to the BlueMix type targets, other applications may be partially transformed (e.g. data stays back at customer's site), other applications may be recommended for rewriting to enable them to take advantage of the cloud native capabilities.



We have identified 4 transformation paths based on the application properties and complexity:

1. Full transformation – application can be transformed and hosted in one of the runtime environments in the target cloud and in addition take advantage of cloud native service management stack
2. Partial transformation – application's core functions are extracted and transformed to cloud native services
3. Guided rewrite – application is analyzed and recommendations are made on how to rewrite it for the target in the microservice architecture style
4. No transformation – for example, application cannot be transformed at all, we can package it as a container and host is as is in the target.

In this paper we present how BlueShift enables the process of full transformation for a sample WebSphere application, PlantsByWebSphere v8.5. It is a standard three tier application, which uses DerbyDB.

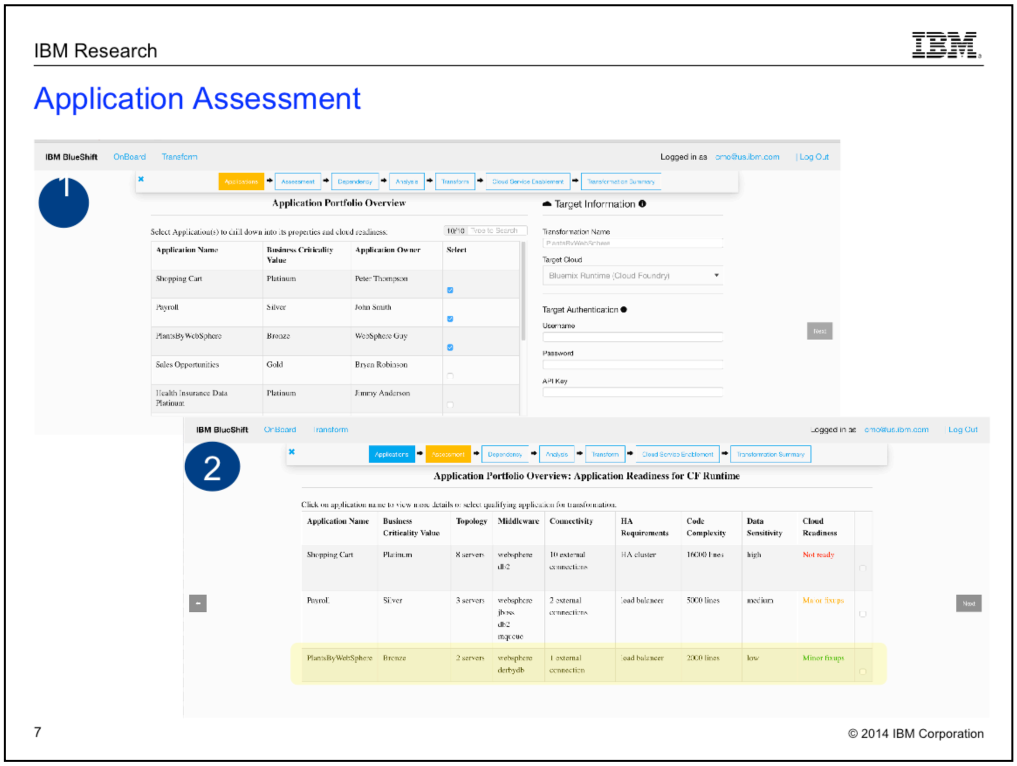
The process of transformation starts with analysis and the assessment of the application's readiness for the target environment. We selected Liberty Runtime in BlueMix as our target.

The process analyzes to which degree the target supports applications functionality and also since we are dealing with a legacy app, which was not written for cloud native environment, where the gaps are.

These gaps are either automatically fixed, where possible (e.g. binding the Session Cache service automatically to overcome issues where application is using HTTPSession – which is not supported in Cloud environment).

For other gaps, developer is given instructions on how to resolve them.

Once the application has been re-bundled is is pushed to the target Cloud, and additional services are bound to it.



At the assessment stage functional and non-functional requirements of applications are collected and used to classify applications based on the level of their complexity and risk associated with the transformation.

Some of the functional requirements include: application topology, middleware type and versions, connectivity (external services, etc., HA requirements

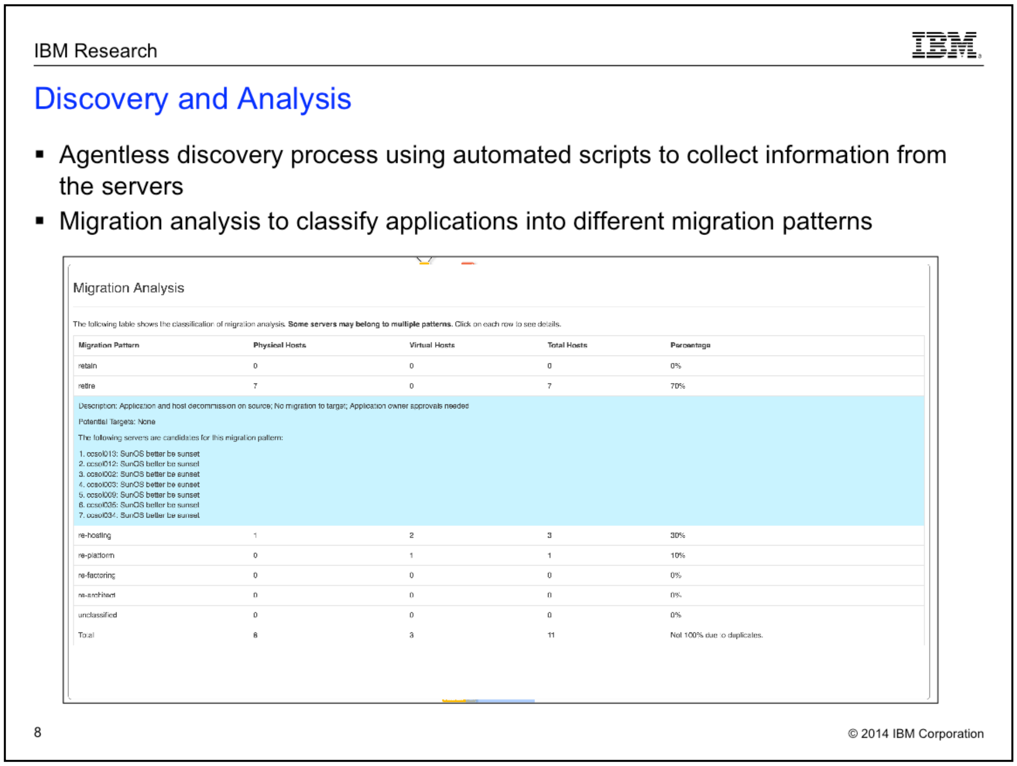
Some examples of non-functional requirements include: licensing, code complexity, business critical value, data sensitivity

This data comes both from automated infrastructure discovery (ref here) and questionnaires distributed to application owners and technical support teams (ref here).

The system currently integrates with ALDM (ref here), and user's also have the possibility of uploading their own data through spreadsheets.

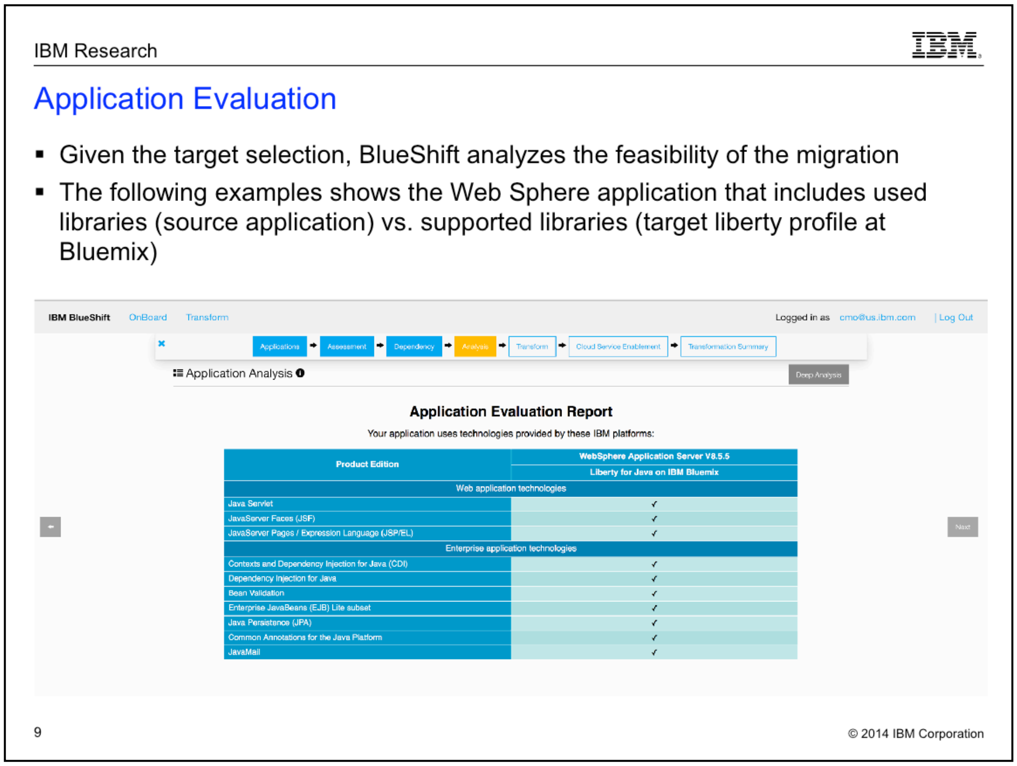
At step 1 user selects a set of applications for transformation to the desired target. Target is selected from the dropdown on right hand side, and target authentication is supplied.

At step 2 user views the report and classification of the applications based on their complexity and selects the application with lowest risk for transformation.



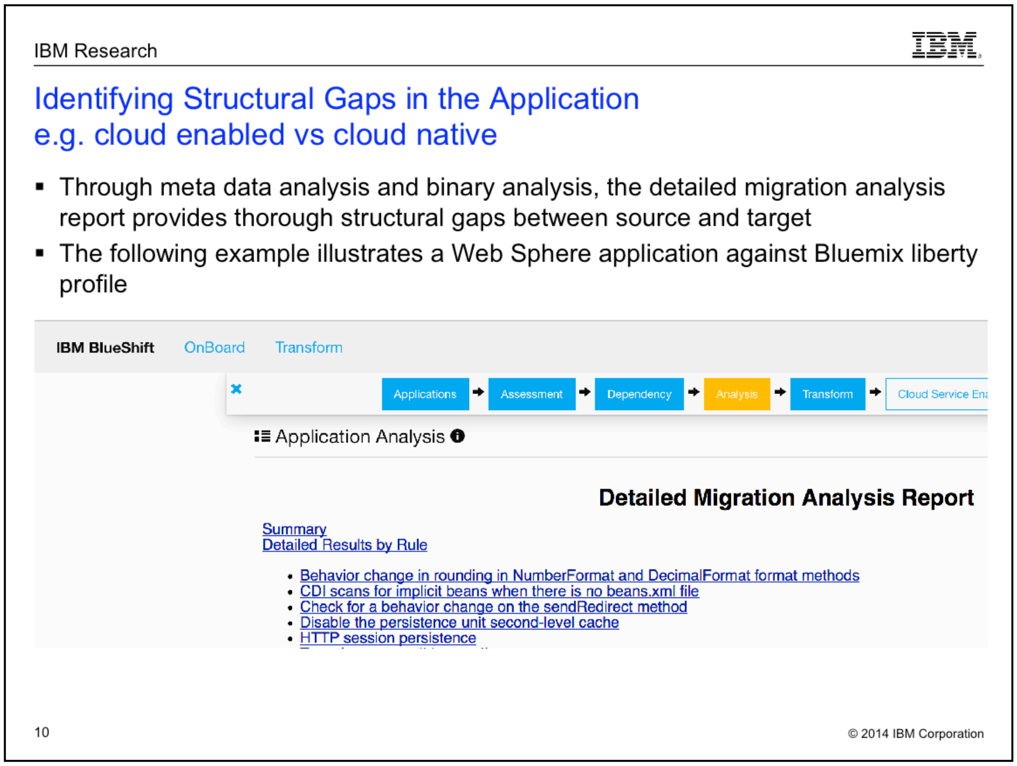
BlueShift takes an agentless discovery approach, using automated scripts to collect information from the servers. Since enterprise customers are very sensitive to security, it is extremely hard to obtain machine credentials without compromising privacy. Thus, BlueShift is deployed into the customer's data center, and isolated from the outside (even from IBM). The collection process can be executed automatically through remote access such as ssh or samba, or alternatively the user can run the scripts themselves and upload the resulting archived files to BlueShift. This process collects data about system properties, CPU/memory/disk/network usages, network statistics, as well as specific services, processes, and applications of each server.

Being able to group servers by a variety of metrics helps users to determine the significance of migrating each application. BlueShift conducts a migration analysis of the architecture, classifying servers into different migration patterns, suggesting whether each could be retained in the current state, retired, re-hosted, re-platformed, re-factored, or re-architectured. Each pattern is a good indication of where the servers migrate into. BlueShift also allows users to cluster the servers in the graph through a variety of statistical algorithms and view each group of servers visually.

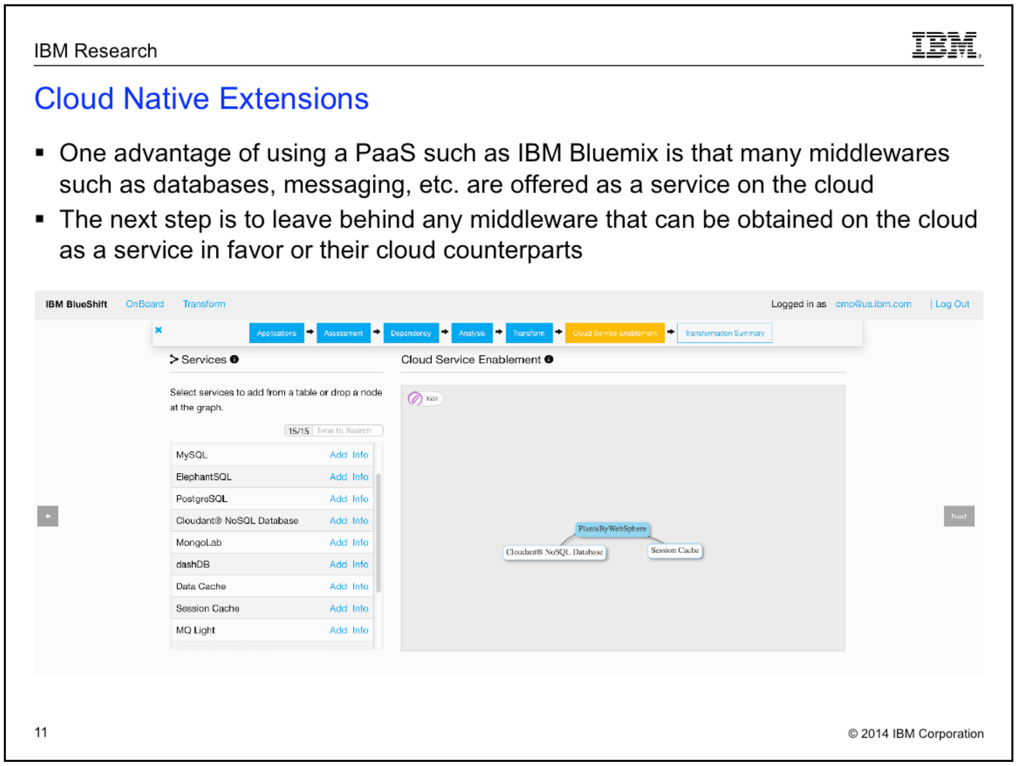


Product Edition	WebSphere Application Server V8.5.5 / Liberty for Java on IBM Bluemix
Web application technologies	
Java Servlet	✓
JavaServer Faces (JSF)	✓
JavaServer Pages / Expression Language (JSP/EL)	✓
Enterprise application technologies	
Contexts and Dependency Injection for Java (CDI)	✓
Dependency Injection for Java	✓
Bean Validation	✓
Enterprise JavaBeans (EJB) Lite subset	✓
Java Persistence (JPA)	✓
Common Annotations for the Java Platform	✓
JavaMail	✓

BlueShift runs a binary scanner to analyze each application in depth. The first application evaluation report with meta-data analysis includes the platform-level analysis that can tell whether or not all the libraries used in the source application can be still used at the target environment. For example, the Bluemix runtime liberty profile provides a group of libraries with certain versions so that it is important to see whether the compatibility requirements are met. Once all the compatibility requirements are met, BlueShift looks at the deep binary analysis (next slides).



BlueShift runs both meta-data analysis and binary analysis, and collects transformation information in order to recommend modifications to the target application. Some examples include the code update requirements such as ”Behavior change in rounding in NumberFormat and DecimalFormat format methods” and “Check for a behavior change on the sendRedirect method”. BlueShift also analyzes service level dependency so that it can replace add up necessary services at the target during the transformation. For example, when an application uses a http persistent cache, it may be recommended to use the http session persistency service at the target such as Bluemix runtime.



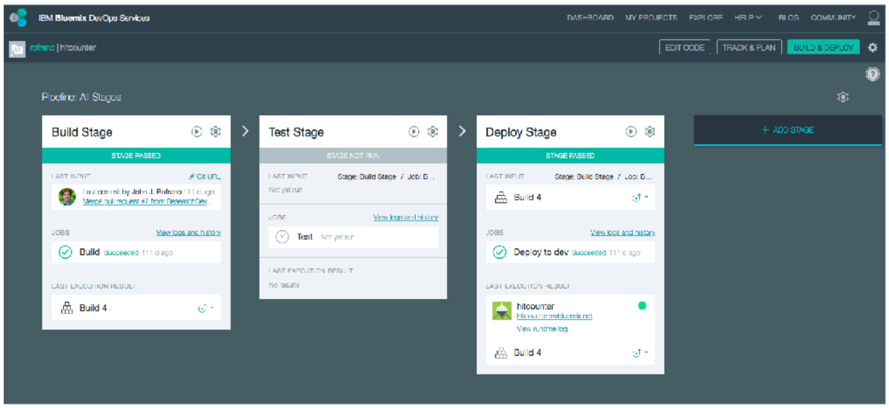
Cloud providers such as IBM Bluemix offer more than just Platform as a Service. They offer middleware services such as databases, messaging, analytics, Watson services, etc. as a service. Applications that are migrating to the cloud should leave behind any middleware they can in favor of using their counterparts in the Cloud. This can be a one-for-one replacement of like software, (e.g., DB2 to DB2) or a similar replacement (e.g., Oracle to PostgreSQL). BlueShift helps you to make these choices and bind the new services to your application.

The objective is to minimize what needs to be maintained in the cloud to just your application and nothing more if possible. This will free up support teams to just support the application rather than managing middleware servers, software version updates, and patches.

IBM Research

Transformation to DevOps

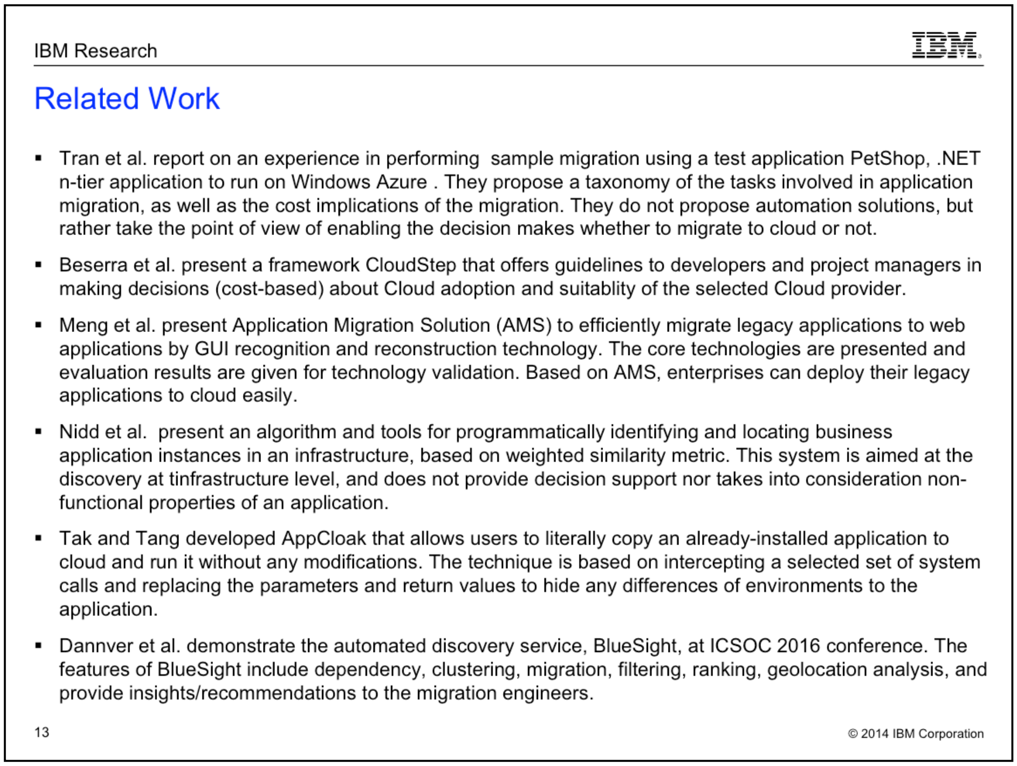
- Migrating an application to a Cloud Native Architecture alone and keeping it's deployment manual is only half of the transformation
- Leveraging DevOps practices and specifically is automated CI/CD Pipeline is the key to maintaining efficiency in the cloud



12

© 2014 IBM Corporation

Critical to the transformation to a Cloud Native Architecture is the adoption of DevOps practices. It is not enough just to get the application into the cloud and leverage cloud native services. You need to be able to redeploy effortlessly it anytime a change is required. Setting up a Continuous Integration and Continuous Deployment Pipeline is the final step in completing the transformation. Without this step you cannot obtain the full benefit of using platform as a service and cloud native technologies. Also critical is the Testing Stage. To gain the benefit of CI/CD you need to automate testing. This is something most legacy applications don't have but is something that needs to be done in order to take advantage of an automated pipeline. This should also prepare application development teams to leverage automated testing as they build future cloud native applications.



Van Tran, Jacky Keung, Anna Liu, and Alan Fekete. 2011. Application migration to cloud: a taxonomy of critical factors. In *Proceedings of the 2nd International Workshop on Software Engineering for Cloud Computing*(SECLOUD '11). ACM, New York, NY, USA, 22-28. DOI=http://dx.doi.org/10.1145/1985500.1985505

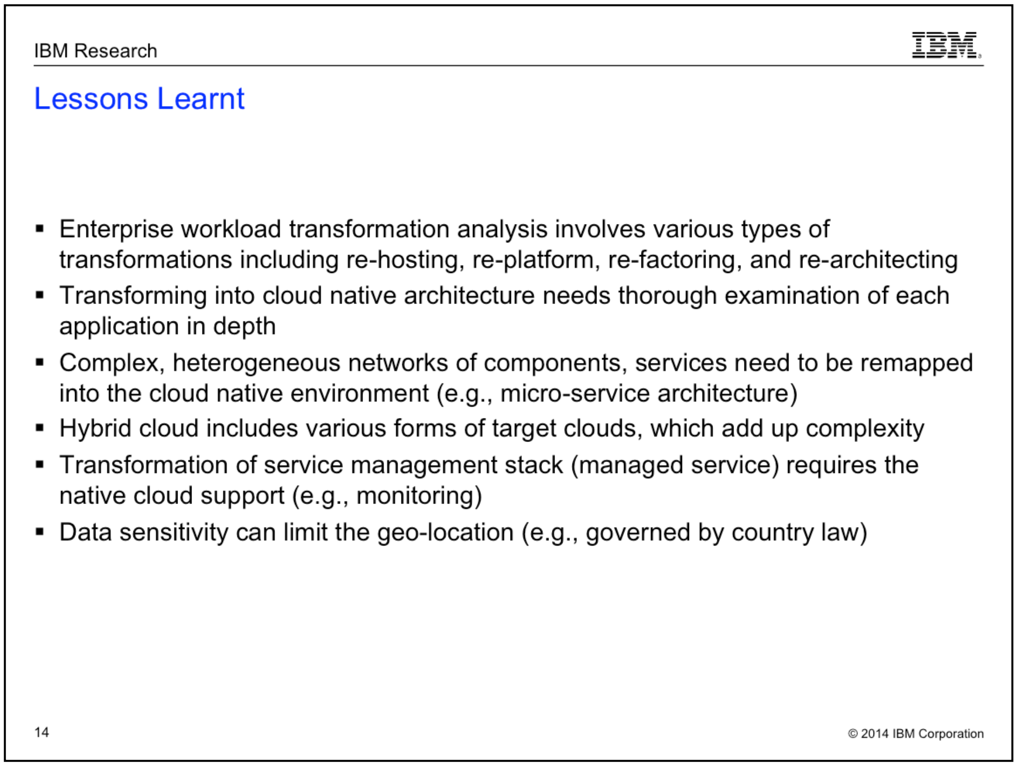
P. V. Beserra, A. Camara, R. Ximenes, A. B. Albuquerque and N. C. Mendonça, "Cloudstep: A step-by-step decision process to support legacy application migration to the cloud," *2012 IEEE 6th International Workshop on the Maintenance and Evolution of Service-Oriented and Cloud-Based Systems (MESOCA)*, Trnto, 2012, pp. 7-16.

X. Meng, J. Shi, X. Liu, H. Liu and L. Wang, "Legacy Application Migration to Cloud," *Cloud Computing (CLOUD), 2011 IEEE International Conference on*, Washington, DC, 2011, pp. 750-751.

B. C. Tak and C. Tang, "AppCloak: Rapid Migration of Legacy Applications into Cloud," *2014 IEEE 7th International Conference on Cloud Computing*, Anchorage, AK, 2014, pp. 810-817.

M. Nidd, K. Bai, J. Hwang, M. Vukovic and M. Tacci, "Automated business application discovery," *2015 IFIP/IEEE International Symposium on Integrated Network Management (IM)*, Ottawa, ON, 2015, pp. 794-797.

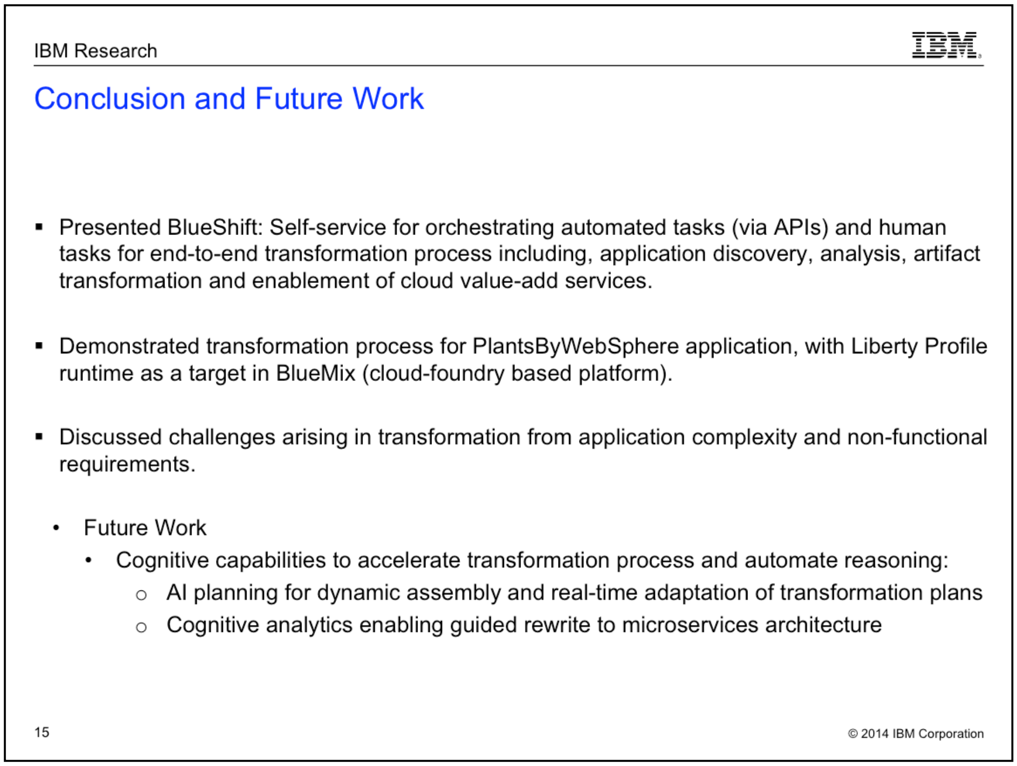
Dannver Wu, Jinho Hwang, Maja Vukovic, Nikos Anerousis, “BlueSight: Automated Discovery Service for Cloud Migration of Enterprises,” International Conference on Service Oriented Computing (ICSOC), Demo Track, 2016



BlueShift takes on the vision to automate the end-to-end transformation process from discovery to testing the transformation, but there are a lot of obstacles we have to address before we can transform with smooth automation. Throughout the real enterprise transformation process, we learned that the enterprise workload transformation analysis involves various types of transformations including re-hosting, re-platform, re-factoring, and re-architecting, and transforming into the cloud native architecture requires thorough examination of individual application to verify the compatibility.

Also, the complexity and heterogeneity of components and services add up more complexity so that remapping the legacy data center applications into the native cloud application becomes challenging. This is absolutely true in the hybrid cloud environments that include various infrastructure service management stack.

Other than technical challenges, the administrative challenges include data sensitivity such as data protection law in each country.



We presented BlueShift -- a service for automated application transformation to a Cloud Native architecture, specifically BlueMix, IBM's cloud platform as a service (PaaS) based on Cloud Foundry open technology.

BlueShift automates an end-to-end transformation process including, application discovery, analysis, artifact transformation and enablement of cloud value-add services. We show how BlueShift automates transformation of Plants by WebSphere application to Liberty Profile Runtime. We further showed setup of SessionCache service, and enablement of an automated pipeline and monitoring through cloud native services for the transformed application. We discussed lessons learnt and challenges encountered during this transformation execution.

Currently our system focuses on automation of the transformation process for applications that fit to the target environment (e.g. version supported, etc.). Our future work is two-fold, we plan to focus on dynamic generation of such transformation plans, due to the high variability and complexity of the applications. There is no one size fits all transformation plans, also configuration of applications and underlying infrastructure may change over the time and plans need to be adapted.

Secondly, we want to extend the platform to support rewrite use cases. That will require access to code and reasoning capabilities on how to enable rewrite of functionality in the cloud native form.