One-click Application Deployment - An Approach for Automated Deployment of Instantiable Cross-platform Mobile Applications

Arnold Arz von Straussenburg
University of Münster
arnold.arz@uni-muenster.de

Friedrich Chasin
University of Cologne
fchasin@uni-koeln.de

Abstract

Deployment of cross-platform mobile applications remains a task almost exclusively performed by application developers. Even with applications that are instantiated multiple times as stand-alone configured versions of a same application for different clients or purposes, the deployment requires organizations to allocate developers' time and know-how to navigate the complex process of submitting application instances to different platforms. We extend the body of knowledge on cross-platform applications, which is currently dominated by literature covering aspects of application development, with a dedicated approach for cross-platform application deployment. Our approach enables non-technical roles in an organization to trigger a 'one-click' workflow for deploying instantiable cross-platform applications and applies to various scenarios in which stand-alone configurations of the same applications are required. The approach spurs academic inquiries into application deployment and has practical implications for organizations that want to streamline their application deployment, reduce required resources, and improve deployment efficiency.

1. Introduction

Cross-platform development is an effective way of addressing mobile application development challenges for different target platforms. These challenges include organizations assigning IT infrastructure, skills, and time resources to each target platform application. However, the heterogeneity of mobile platforms brings challenges beyond application development. In contrast to the well-research aspect of cross-platform development [1, 2, 3, 4], little methodical guidance exists for cross-platform application deployment. Accordingly, deployment of cross-platform applications continues to be a task performed for each platform independently and requiring the involvement of application developers. This challenge can be multiplied in situations where organizations require releasing similar stand-alone applications. In these scenarios, data, access, and functionalities are separated, leading to multiple instances of the mobile applications being generated from the same source code and configured for specific user groups. Each instance of such applications represents a stand-alone application and requires individual deployment. One example is a car-sharing IT platform, which allows third-party car-sharing providers to launch their own independent applications for the system [5].

Multiple reasons exist to create instantiable cross-platform applications. First, with multi-tenant solutions such as the shopping platform Salesforce or the customer relationship management platform HubSpot, representing a typical solution for separating business client content, clients can find their corporate identity insufficiently reflected. Second, business customer organizations can be interested in controlling the deployment process themselves to maintain metadata, screenshots, customer reviews, and descriptive text that can inform end-users and lead to more downloads. Third, individual instances can be subject to continuous feature development and instance-specific updates. In multiple instances applications, requirements for the publishing, deployment, and maintenance process increase with each additional instance of a released mobile application. This requires technically skilled developers to be involved in the deployment process. Against this background, organizations that require the creation of multiple stand-alone cross-platform mobile apps that represent a common set of features currently have no practical way to reduce their deployment overhead. Addressing this problem, this work proposes an approach that enables non-technical roles in an organization to perform a one-click workflow to the extent possible to deploy instantiable cross-platform applications.

The remainder of the manuscript is structured as follows. In the next chapter, we apply both

URI: https://hdl.handle.net/10125/80274
978-0-9981331-5-7

academic and industrial lenses to examine the domain of cross-platform applications. The chapter provides background on the practice and research in the area of cross-platform application development and deployment. The chapter 3 focuses on the objectives of the design-oriented approach taken in this work to develop the deployment approach. Subsequent chapters mirror the steps of the research approach and elaborate upon the design (chapter 4) and the demonstration (chapter 5) of the proposed deployment approach. We conclude with a discussion of the developed artifact in chapter 6, including argumentative evaluation of the approach against the background of deployment approaches in academia and practice, the limitations of the approach, and avenues for further research.

2. Background

The mobile operating system market is currently divided into two main operating systems, Apple's iOS and Google's Android, henceforth referred to as mobile platforms. These platforms come with differences in development, compiling techniques, different development approaches, and developer platforms, as well as significant differences in platform Application Programming Interface (API). For these platforms, native applications are developed in a platform-specific Integrated Development Environment (IDE) like XCode and Android Studio. In addition, different programming languages are employed. These are, for instance, Kotlin or Java for Android and Swift or Objective-C for iOS. Therefore, an application aimed at customers across different platforms must be developed from scratch to support each new platform [2]. To counter this duplication of development effort, organizations create cross-platform applications that run on multiple mobile platforms according to the Write Once, Run Anywhere (WORA) principle.

Various approaches to developing cross-platform applications exist, as well as a large number of associated frameworks [6, 7, 8]. These approaches are differentiated from each other by distinguishable characteristics of the development frameworks and include hybrid applications, which in essence provide a WebView wrapper in a native application, interpreted applications, which can execute interpreted scripts written in for instance JavaScript, and cross-compiled applications, which compile native code for different platforms from a common code base [2, 1]. There are also model-driven approaches that follow various code generation approaches in order to create cross-platform apps [9, 10]. In addition, there are Progressive Web App (PWA) [11], which have web applications with additional functionality to run in an offline environment and are typically not deployed through the app stores. However, the naming of the groupings of frameworks may differ [12]. The field of cross-platform development is rapidly evolving and attracting research attention [1, 2, 3, 4]. As a result, new technologies and approaches are constantly being developed, and new development frameworks and approaches, which aim to achieve a native user experience, have become increasingly popular in recent years [11]. Alongside the development of the application, deployment is also an elementary step in the provisioning of mobile applications [13]. Development and deployment of an application are two distinct activities differentiated from each other in the following. Creating the source code or binary files of the application is only the first step of a successful application provision, regardless of the development approach used. Registering an application on the developer platforms of Apple and Google, publishing the beta and production versions of the application, and maintaining metadata for the app stores are further steps that must follow development. The same applies to the subsequent updating of the application via the application lifecycle. These different phases of deployment are shown in figure 1.

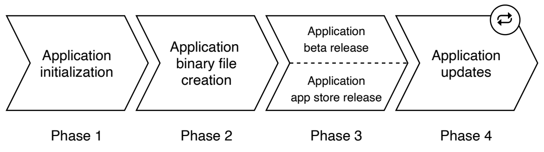
Application initialization
Application binary file creation
Application beta release
Application app store release
Application updates
Phase 1
Phase 2
Phase 3
Phase 4

Figure 1. Deployment phases.

The challenge of making applications available for different mobile platforms arises in the deployment and the development of the applications. The goal of deploying a mobile application via the Apple and Google app stores is a major challenge for developers because of the different ecosystems of the platforms and the different deployment approaches [14]. While in cross-platform development, the different approaches allow developing a common code base for both platforms [2], there is no comparable solution for the deployment of applications. Often the deployment of applications on the different platforms is a task that has to be done manually by a developer for the platforms separately. This approach can become a problem when an organization wants to deploy many applications on different platforms.

No dedicated research on cross-platform mobile application deployment could be identified in the literature. However, three different fields that can be considered related fields to the deployment of cross-platform applications and have been studied more intensively are the development of cross-platform applications, the deployment of mobile applications that have not been developed in a cross-platform context, and the deployment of software in a general context. In contrast to the large body of research concerned with the first related field to cross-platform mobile app deployment, the development of cross-platform mobile applications [2, 15, 16, 17, 18], there is no comparable body of research on the deployment of cross-platform applications to different platforms.

In the literature, deployment costs and time-to-market are recognized as key factors in cross-platform mobile development [19]. Nevertheless, deployment is only seen as a factor for choosing the cross-platform approach to develop the application despite its importance. Existing research that focuses on cross-platform mobile application development treats deployment as an afterthought [20]. The lack of a holistic view on deployment can be seen in surveys that address deployment as an aspect of cross-platform development focusing only on generating application binaries using various cross-platform frameworks [21]. The subsequent deployment of generated binaries to the app stores has not been under similar academic scrutiny. Challenges of deployment are broken down to establishing deployment mechanisms, like the platform-independent deployment of web applications or platform-dependent deployment using, for instance, Mobile Enterprise Application Platforms [22].

The distinction between deployments to app stores, enterprise deployments, or ad-hoc deployments is recognized [22], the additional complexity caused by deployment via the app stores is seen as a disadvantage for cross-platform applications [14]. Deployment to the app store of the respective platform is frowned upon in certain contexts in the case of web-based cross-platform approaches [14]. The argument is made, that cross-platform applications tend to get a lower prioritization in the recommendations in the app stores [14]. Additionally, instances arose where cross-platform applications that only deliver Hypertext Markup Language (HTML) and JavaScript code in a thin native container, such as in the case of many hybrid applications, were outright rejected during the review process by the platform providers Apple and Google [14] and consequently published via the app stores. This is one reason why applications built with platform-specific, native code still dominate the app stores [23]. Instead, ways to deploy cross-platform applications via means other than the app stores are explored [24]. One possibility is the deployment of PWAs on web servers [11]. The latter's advantages include faster deployment without review procedures, no installation or updating of applications via an app store, and guaranteed compatibility to future mobile devices and operating systems [24]. Moreover, it allows access from any computer with a modern web browser, and the standardized web technologies enable compatibility with a large set of mobile and non-mobile devices [24]. The continuous evolution of web technologies, such as the emergence of WebAR, also increasingly allows complex mobile applications with features such as Augmented Reality (AR) to be enabled by mobile browsers [25, 26]. However, these alternative deployment methods are not always viable alternatives for organizations that rely on deployment through the app stores since PWAs still do offer fewer possibilities to the application developers than other cross-platform approaches [25].

While in the research on the development of mobile cross-platform applications, the deployment part of providing mobile applications is largely ignored, it is mentioned in the context of non-cross-platform applications [27], a second related field of research. Deployment has been subject to research in different single instance application contexts, e.g., highlighting challenges of deployment and centralized and decentralized frameworks [27] or the Continuous Deployment (CD) of large applications. Still, in the context of single instance applications, a difference remains in the research on mobile application development and deployment. Outside the mobile application context, there is a larger amount of research, primarily from the software engineering field, on the topic of software deployment and CD. The literature addresses the challenges of adopting CD in practice [28]. These include testing and integration problems, which are most common, but also building, designing, releasing software, and human and organizational resources of an organization [28, 29].

3. Design objectives

This work is based on Design Science Research (DSR). The objective is to create innovative solutions to new and unsolved problems originating in practice [30]. For this purpose, a Design Science Reserach Methodology (DSRM) [31] is employed to develop an approach for deploying instantiations of instantiable mobile applications.

The overall design objective is to develop a deployment approach that facilitates organizations configuring and deploying multiple mobile applications. The overall objective yields four operational sub-objectives, which arise from the challenges from the industry in practice and which an approach for deployment must fulfill: The first objective is the repeatability of the approach to a high degree, the second one is the deployability for different deployment platforms, relieving technical members of the organization is the third objective. The fourth objective is enabling non-technical members of the organization to perform a deployment. The latter two objectives arise from the organizational perspective on the deployment process and are necessary to enable the high repeatability of the deployment. By fulfilling these objectives, time and resources spent on creating different applications that are functionally similar can be reduced.

For the design objective of developing an approach that supports organizations in deploying multiple instances of mobile applications, repeatability is an essential component and basis for implementation to create many different instances reliably. Secondly, a cross-platform approach is needed for the deployment approach to reach different customers from different mobile platforms equally. Otherwise, a situation arises where the deployment for one platform differs from the deployment for another platform, whereby the advantages of a joint, comprehensive deployment approach are missed. To achieve the first objectives for the deployment approach, especially for the high repeatability, it is necessary to reduce the responsibility of technical members for the deployment. The time and resources of application developers are traditionally the limiting factors in the deployment of mobile applications. Thus, the involvement of technical members of the organization, who are usually responsible for the deployment of software products, is to be reduced. After an initial setup of an instantiable base application and an automated deployment pipeline, resources of the technical members of the organization should be freed, which otherwise would have been committed in the deployment. Enabling non-technical members of the organization to trigger the deployment of new instances of an instantiable mobile application and configure them is fundamental to relieving the technical members of the organization. The objective here is to enable a one-click deployment for applications as far as possible.

4. Design

4.1. Organizational perspective

A high-level overview of the traditional approach to application development in organizations is shown in figure 2 to position the approach in the state-of-the-art deployment. Basic activities that need to be performed to create a new application and by whom they are performed are shown. Here, activities are classified according to whether non-technical or technical organization members perform them. The most critical activities in creating a new application include conceptualizing the application and communicating with the technical members of the organization. On the technical side, a sequence of manual activities commences, starting with creating a new application base. Even when creating a new application with minimal new functionality using an existing application template, manual configuration and possibly feature development ensues. Subsequently, the application is compiled by the developer and deployed on the various platforms of the platform operators. The application developer is also responsible for creating application records on the app store operator's developer platform and managing application metadata. Finally, the deployed application can be reviewed and tested by the requesting non-technical members to be published publicly afterward. Even if the new application has no new functionality compared to the organization's existing applications, this represents a significant effort for the technical members. However, non-technical members already took on determining the functionality of the application in the first step. Yet, that in itself has traditionally not been enough to carry out the creation of a new application.

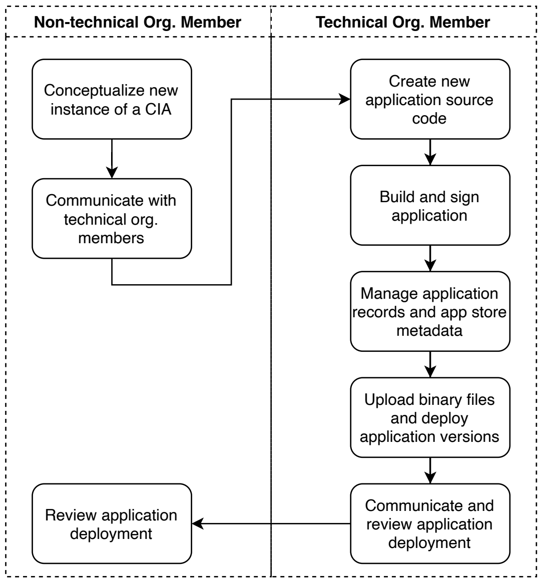
Non-technical Org. Member
Technical Org. Member
Conceptualize new instance of a CIA
Communicate with technical org. members
Create new application source code
Build and sign application
Manage application records and app store metadata
Upload binary files and deploy application versions
Communicate and review application deployment
Review application deployment

Figure 2. Traditional deployment approach.

Comparing to the traditional approach, the new approach proposed here is depicted in figure 3. The new approach also starts with the conceptualization of the latest application by the responsible members of the organization, who do not necessarily have to be technically skilled. However, the tasks of the technical members of the organization have been replaced by an automated system, which takes over the creation of the new application and can then deploy it independently. Thus, anyone without further technical skills can initiate the deployment. Of course, this requires a valid configuration and a unique type of application that can be created, a particular kind of basic application that members of the organization can deploy as autonomously as possible to relieve technical members of the organization of the deployment tasks, which henceforth will be called a Configurable Instantiable Application (CIA). Any new applications that provide a subset of the functionality of the base application and can be configured in their functionality and appearance are called instances of the CIA. This base application can be configured, built, and deployed through an automated pipeline. Given that such an initial base application has been created and an automated deployment pipeline has been established, an arbitrary number of new instances of the can be created, configured, and deployed by any member of the organization. This provides a solution to the problem of deploying multiple feature-like applications with limited technical resources.

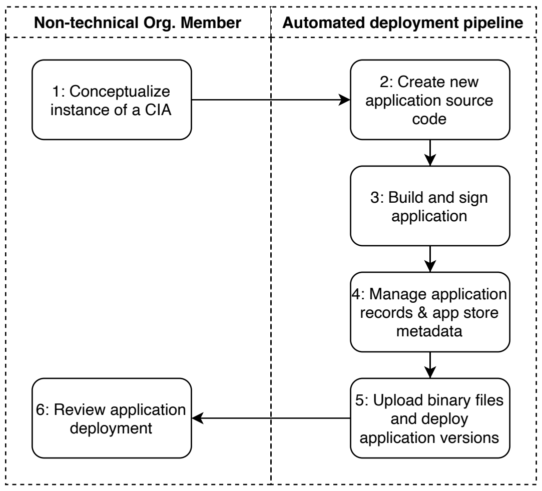
Non-technical Org. Member
Automated deployment pipeline
1: Conceptualize instance of a CIA
2: Create new application source code
3: Build and sign application
4: Manage application records & app store metadata
5: Upload binary files and deploy application versions
6: Review application deployment

Figure 3. Deployment approach for CIA.

Using the approach presented here, non-technical members of the organization still solely conceptualize the application and then press a button to trigger a build. Instead of time-consuming communication with the technical members of the organization, the configuration is also performed directly by the non-technical members. The reduced communication overhead can thus effectively reduce the effort of the non-technical members in deploying a new application. With the new approach, developers are no longer involved in creating a new instance of CIA. Of course, it is within their domain to initially develop an CIA, as well as the capability to configure it. The deployment pipeline, which is supposed to take over the developers' tasks, is also set up and adapted by the developers. However, the initial setup effort and the subsequent time savings are positively offset when many instances of a CIA are created.

4.2. Technical perspective

A more technical explanation of how organizations in practice can realize the approach proposed here for deploying mobile applications is given to provide a tangible understanding. An Unified Modeling Language (UML) sequence diagram-like depictions with labeled lifelines illustrate the process of deployment.

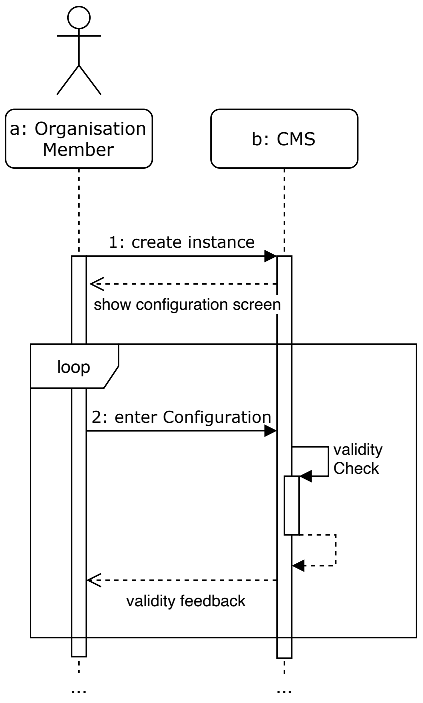

Figure 4. Create a configuration in a Content Management System (CMS).

Conceptualization of an instance of a CIA by the responsible members of the organization is the first step for its creation. To manage the various configurations of instances of the CIA, a CMS can be utilized. This is illustrated in figure 4. This way, the creation of a new instance of the application is realized by creating a new entry in the CMS, by making the necessary configurations therein. A validity check of the configuration by the CMS supports non-technical members of the organization only to create configurations that can be further used in the following activities.

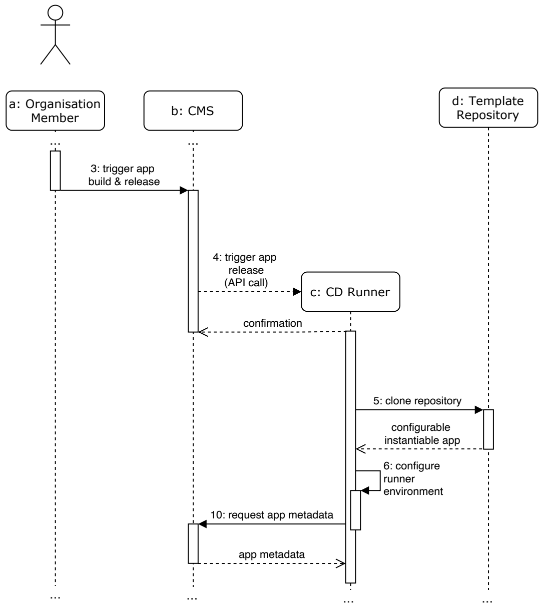

Figure 5. Triggering the application creation.

With a valid configuration for an application created in the CMS, members of the organization can trigger the creation of the application. A one-click interaction with the CMS triggers an automated deployment pipeline. Upon triggering the creation of the application, a new deployment is started through an API call from the CMS to a server that runs instances of the automated deployment pipeline, called the CD runner. The API call to the CD runner provides the basic configurations to be performed on the new instance by the deployment pipeline. At this point, the deployment pipeline takes over the critical activities necessary to create and deploy a new instance of CIA, independently and without intervention from a technical member of the organization. For this purpose, the base application of CIA, the template, is copied from a central code repository. This template consists of a common code base, from which the deployment pipeline can generate applications for the different mobile platforms according to the cross-platform approach. Depending on the platform for which the app is created and deployed, initial configurations are carried out on the runner's virtual environment. Subsequently, additional information like metadata for the app stores is retrieved via an interface provided by CMS for the deployment. These activities are illustrated in figure 5.

Using the configuration retrieved from the CMS, the application template of the CIA is configured afterward. One way of configuring the CIA is to modify the source code of the application using configuration parameters. This configuration distinguishes different instances of CIA from each other. In the next step, a central repository that holds developer certificates for signing mobile application code provides the required signing data. Once the application configuration is complete and signing certificates are obtained, the application is compiled and signed for the mobile platform. Provided that the mobile platform provider's developer platform supports the automated creation of application records for deployment via the app stores through an API, this step follows based on the previously obtained application metadata from the CMS. These activities are depicted in figure 6.

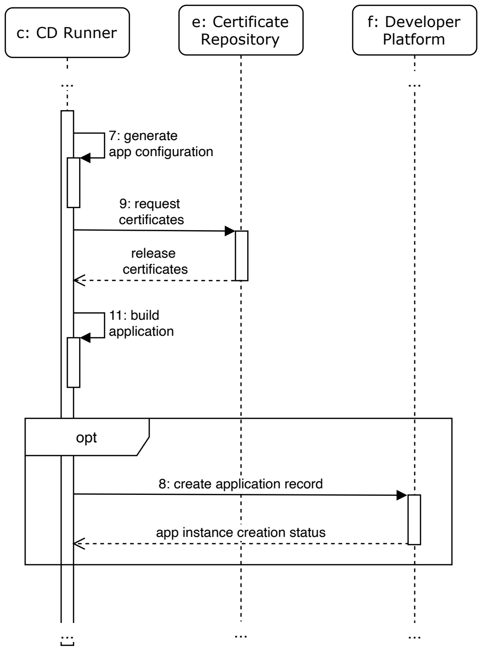
c: CD Runner
e: Certificate Repository
f: Developer Platform
7: generate app configuration
9: request certificates
release certificates
11: build application
opt
8: create application record
app instance creation status

Figure 6. Compiling and signing the application code.

As soon as the compiled application has been created and an application record exists on the developer platforms of the mobile platform providers, the application is uploaded to the mobile platform provider's development platform. The application can now be made available as a beta release, which can then be tested and tried out. Alternatively, after testing, the application may be released to be published to the app store through the app store provider's review process. These activities are shown in figure 7.

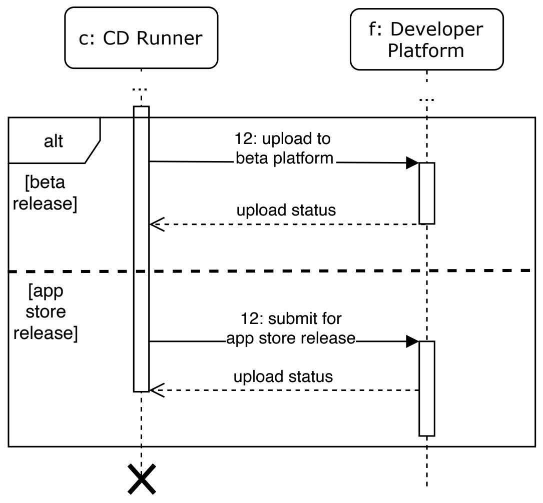

Figure 7. Deployment of the application for mobile platforms.

App No.	No. clicks	Available Features	Platform
1	<50	Event schedule, speaker information, downloads	Android
2	1	Event schedule, speaker information, venue map, notifications screen, external links, custom HTML page	iOS
3	>100 (manual)	Event schedule, speaker information, downloads	iOS

Table 1. App instantiations.

5. Demonstration

In the following, the concept of CIA and the approach to deploy it are demonstrated. For this purpose, a series of applications were created for the event industry. These are built based on an existing application in the industry and provide the same functionality as the application that has been already in use. It was completely redeveloped as a CIA to allow for the deployment approach. The event industry is particularly suited for demonstrating CIA. Event organizers typically organize multiple events. These may be very similar to one another or completely different and may occur on a regular or irregular basis. Multiple events organized by one organizer can even conceivably take place in parallel. Some events benefit from a supporting application that digitally prepares information for the participants. However, the requirements for the applications supporting these events are as varied as the events themselves. Providing different applications for several events usually poses a great difficulty for the organizers. In addition, managing data and information about their events is a core competency of organizers. But technical capacity for developing and deploying applications can be limited. The event application demonstrated here allows participants at digital, hybrid, or physical events to find out information about the schedule, location and rooms, speakers and presenters, the event itself and more. Moreover, the application can be customized not only in its appearance but also in the functionalities it offers. For demonstration purposes, two different instantiations of a common base application have been created, pictured in figure 8. The two instantiations have various features of the base application enabled. The first application is an elementary event application, which can only display basic information about the event, the schedule, speakers, and available downloadable documents. The second application is a more feature-rich application, which offers information about the event, schedule, and speakers and includes an included map of the venue and a notifications page. Beyond the enabled features of the base application, the second application includes additional content pages that display custom HTML content and a link to an external web page. The available features are shown on a start screen as menu buttons. Both applications were automatically instantiated with a single click after configuration in an event management system and deployed as a beta version. But since Google does not support creating application records on their developer platform, Google Play Console, this step still had to be performed manually for the first application. This resulted in a more complex deployment than the one-click solution for the iOS application. The second application was also once created and deployed manually by a developer. This is a more complex endeavor in which the application must be manually created and configured. More manual interaction is then required with the developer platforms of the platform providers. An overview of the application features and mouse clicks required for deploying the application is shown in table 1. These deployments were performed on the iOS and Android platforms, respectively. The number of clicks required is used here as an indicator of the complexity of the task. We can confidently report that non-technical members can create a new instance of the application via the automated approach. This is not the case with the manual approach of the third application, which requires a considerably higher number of interactions with both the IDE and the development platforms.

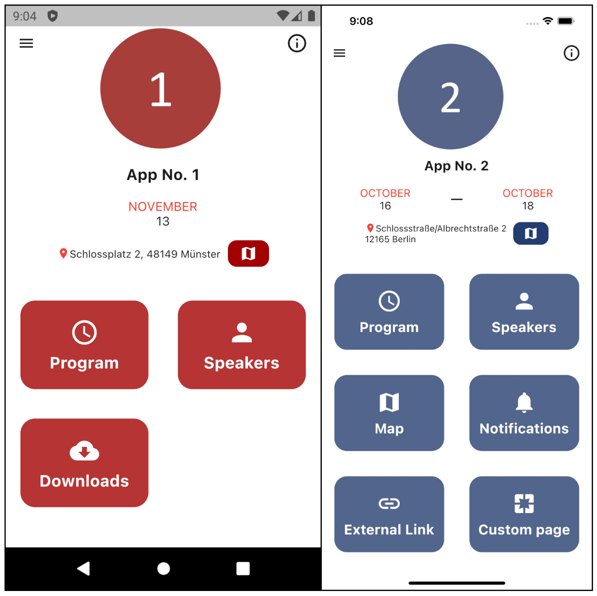

Figure 8. Different configurations of the event application.

The event management system Indico[1] is used to manage the data for the various events. Using the system, organization members can create data entries for events, and enter information about schedules and speakers, among other details. Furthermore, a plugin that was developed specifically to this end can be used to manage configurations about the appearance and functionality of an application associated with the event and metadata for the app stores directly in Indico. Information about the event, as well as configuration parameters of the application, are then available via an API of the Indico system. Once the event has been created in the event management system, and the application has been configured, an automated process, the deployment pipeline, can be triggered at the push of a button to create the application and deploy it as a beta version to the beta platforms of the app stores.

The application created in this project is based on the cross-platform framework Flutter, which follows the cross-compilation approach [2]. The choice of the framework makes it easier to compile applications for both iOS and Android based on one codebase written in the programming language Dart, which can be deployed via the app stores. The CIA approach is implemented by creating a base application that has the ability to retrieve design elements such as logos and colors at application run-time from the API provided by Indico. By a similar approach, features of the base application are also enabled or disabled in an instantiation by the application retrieving a configuration at run-time via the Indico API. The base application has a variety of functions that are not required by every instantiation. The configuration for an instantiation from the Indico API thus affects the menu items displayed in the application and the functionalities a user can access. A unique event identifier is written to each instance in the source code at compile-time to enable the retrieval of dynamic contents at application run-time. This ensures that an application retrieves configurations from the API only for the associated event.

Creating an instantiation of the base application at the push of a button is possible for non-technical members of the organization. For this purpose, the processes of creating an instance, configuring it, and then deploying it were automated as far as possible. The automation pipeline is based on the automation framework fastlane. When a new instantiation is triggered in the Indico system, a Hypertext Transfer Protocol (HTTP) call is made to a build server, which then executes the pipeline on the Mac hardware necessary for building iOS applications. The pipeline, which is executed on the remote build servers, consists of several steps: Starting the project and building the application, configuring the application, compiling it, and publishing it across beta platforms. The first step involves copying the code for the base application from the version control system. An application record, is also created on the development platforms of the platform providers Apple and Google. In the second step, the identifier for the event and information such as the application name and application logo is defined. Also, the metadata for the app stores is retrieved from the Indico API through the pipeline. Subsequently, in the third step, the application is compiled by the build server to create the binary files which can be deployed. In the fourth and final step, developer certificates are used automatically to sign the binary files. The application is uploaded to the developer platforms along with the metadata for the app stores. Since the exact implementation of these steps differs between Apple and Google platforms, the pipeline is implemented for both platforms and is executed once for each. A subsequent submission for the platform operators' review process for publication via the app stores can consequently be triggered manually.

6. Discussion and conclusion

In this paper, we developed an approach to enable the repeated deployment of instantiations of CIA. Our approach enabled non-technical roles in an organization to trigger the deployment of these instantiations through

[1]https://indico.cern.ch/

a ’one-click’ workflow. This reduces the responsibility of the technical members of the organization for the deployment activities. Our approach contributes to the body of knowledge in application deployment and, more specifically, cross-platform mobile applications deployment. This area of research has not been studied with an emphasis in the past.

The objectives of this approach were firstly the high repeatability of the deployment, secondly the possibility to perform the approach in a cross-platform manner, thirdly a relief of the technical members of the organization from the activities of the deployment and fourthly enabling of non-technical members of the organization to trigger a deployment independently. We addressed the first repeatability objective by automating a build and deployment pipeline. With this setup, instantiations based on a given configuration can be reliably created, with new configurations. Furthermore, the deployment of an app with the same configuration can be repeated arbitrarily. Different instantiations can even be executed in parallel. The limiting factor for this is only the technical capacity of the build CD runner.

The second objective of deployability on multiple mobile platforms was demonstrated for the two dominant mobile platforms, iOS, and Android. Demonstration on additional mobile platforms was foregone due to the dominance of the predominant two platforms. Since there are significant differences in implementing the activities required for deployment to the respective app stores of Apple and Google, the deployment pipeline needed to be adapted accordingly. In subsequent works, the applicability of the approach for the web platform could be investigated. This would make the deployed application available to even more user groups.

Reducing the responsibility of the technical members of the organization, the third objective, is achieved with the deployment approach to the extent that a new instantiation with a valid configuration no longer requires any intervention by these members. However, this comes at the price of a significant initial resource investment for creating the CIA base application and setting up the automated deployment pipeline. Moreover, interventions of the technical members may still be needed for bug fixing and continuous development of the deployment solution. The limitation of our approach to the initial deployment of a new instance of a CIA does not consider other lifecycle events of an application, such as updating. For these, the approach presented here does not yet provide any relief for the technical members of an organization. Provided that a sufficiently large number of deployments of new instantiations of an CIA is carried out, the technical members will benefit from a net reduction in their workload despite the initial efforts. The freed resources can now be used for activities such as developing new features instead of for deployment. Similarly, the fourth objective, enabling non-technical members of the organization, was achieved under certain conditions. Configuration management requires the ability of non-technical members to create valid configurations. This can be challenging depending on the complexity of the configurations and level of abstraction from the technical implementation in the subsequent build and deployment pipeline. Starting an automated pipeline with a single mouse click minimizes the complexity of triggering the deployment. However, we were only able to implement this approach for the iOS platform. For the Android platform, it is not possible to programmatically create application records in the Google Play Console developer platform via an API. This step still has to be performed manually even with our deployment approach, which again increases the complexity of this approach.

Our approach has practical implications for organizations that want to streamline their application deployment, reduce the required resources, and increase deployment efficiency.

References

[1] A. Charland and B. LeRoux, “Mobile Application Development: Web vs. Native,” *Communications of the ACM*, vol. 54, pp. 49–53, 5 2011.

[2] A. Biørn-Hansen, T. M. Grønli, and G. Ghinea, “A Survey and Taxonomy of Core Concepts and Research Challenges in Cross-platform Mobile Development,” *ACM Computing Surveys*, vol. 51, pp. 1–34, 1 2019.

[3] W. S. El-Kassas, B. A. Abdullah, A. H. Yousef, and A. Wahba, “ICPMD: Integrated Cross-Platform Mobile Development Solution,” in *Proceedings of 2014 9th IEEE International Conference on Computer Engineering and Systems, ICCES 2014*, pp. 307–317, Institute of Electrical and Electronics Engineers Inc., 2 2014.

[4] H. Heitkötter, S. Hanschke, and T. A. Majchrzak, “Comparing Cross-platform Development Approaches for Mobile Applications,” *WEBIST 2012 - Proceedings of the 8th International Conference on Web Information Systems and Technologies*, pp. 299–311, 2012.

[5] P. Ringeisen and R. Goecke, “Flinkster: The Carsharing Platform of Deutsche Bahn AG,” pp. 383–391, Springer, Berlin, Heidelberg, 2016.

[6] M. Latif, Y. Lakhrissi, E. H. Nfaoui, and N. Es-Sbai, “Cross Platform Approach for Mobile Application Development: A Survey,” in *2016 International Conference on Information Technology for Organizations Development, IT4OD 2016*, pp. 1–5, Institute of Electrical and Electronics Engineers Inc., 5 2016.

[7] V. Ahti, S. Hyrynsalmi, and O. Nevalainen, “An Evaluation Framework for Cross-platform Mobile

App Development Tools: A Case Analysis of Adobe PhoneGap Framework," in *ACM International Conference Proceeding Series*, vol. 1164, (New York, NY, USA), pp. 41–48, Association for Computing Machinery, 6 2016.

[8] C. Rieger and T. A. Majchrzak, "Weighted Evaluation Framework for Cross-platform App Development Approaches," in *Lecture Notes in Business Information Processing*, vol. 264, pp. 18–39, Springer Verlag, 2016.

[9] E. Umuhoza, H. Ed-Douibi, M. Brambilla, J. Cabot, and A. Bongio, "Automatic Code Generation for Cross-platform, Multi-device Mobile Apps: Some Reflections from an Industrial Experience," in *MobileDeLi 2015 - Proceedings of the 3rd International Workshop on Mobile Development Lifecycle*, (New York, NY, USA), pp. 37–44, Association for Computing Machinery, Inc, 10 2015.

[10] C. Rieger, "Evaluating a Graphical Model-Driven Approach to Codeless Business App Development," in *Proceedings of the 51st Hawaii International Conference on System Sciences*, Hawaii International Conference on System Sciences, 2018.

[11] A. Biørn-Hansen, T. A. Majchrzak, and T. M. Grønli, "Progressive Web Apps: The Possible Web-native Unifier for Mobile Development," *WEBIST 2017 - Proceedings of the 13th International Conference on Web Information Systems and Technologies*, pp. 344–351, 2017.

[12] T. Majchrzak and T.-M. Grønli, "Comprehensive Analysis of Innovative Cross-Platform App Development Frameworks," in *Proceedings of the 50th Hawaii International Conference on System Sciences (2017)*, Hawaii International Conference on System Sciences, 2017.

[13] F. Chen, J. Zhou, X. Xia, H. Jin, and Q. He, "Optimal Application Deployment in Mobile Edge Computing Environment," in *IEEE International Conference on Cloud Computing, CLOUD*, vol. 2020-October, pp. 184–192, IEEE Computer Society, 10 2020.

[14] K. Shah, H. Sinha, and P. Mishra, "Analysis of Cross-Platform Mobile App Development Tools," in *2019 IEEE 5th International Conference for Convergence in Technology, I2CT 2019*, Institute of Electrical and Electronics Engineers Inc., 3 2019.

[15] A. Biørn-Hansen, C. Rieger, T. M. Grønli, T. A. Majchrzak, and G. Ghinea, "An Empirical Investigation of Performance Overhead in Cross-platform Mobile Development Frameworks," *Empirical Software Engineering*, vol. 25, pp. 2997–3040, 7 2020.

[16] C. Rieger and T. A. Majchrzak, "Towards the Definitive Evaluation Framework for Cross-platform App Development Approaches," *Journal of Systems and Software*, vol. 153, pp. 175–199, 7 2019.

[17] A. Biørn-Hansen, T. M. Grønli, G. Ghinea, and S. Alouneh, "An Empirical Study of Cross-Platform Mobile Development in Industry," *Wireless Communications and Mobile Computing*, vol. 2019, 2019.

[18] S. Xanthopoulos and S. Xinogalos, *A Comparative Analysis of Cross-platform Development Approaches for Mobile Applications*. 2013.

[19] A. Sommer and S. Krusche, "Evaluation of Cross-platform Frameworks for Mobile Applications," tech. rep., 2013.

[20] X. Mao and J. Xin, "Developing Cross-platform Mobile and Web Apps," tech. rep., 2014.

[21] I. Dalmasso, S. K. Datta, C. Bonnet, and N. Nikaein, "Survey, Comparison and Evaluation of Cross Platform Mobile Application Development Tools," in *2013 9th International Wireless Communications and Mobile Computing Conference, IWCMC 2013*, pp. 323–328, 2013.

[22] M. Popa, "Considerations Regarding the Cross-platform Mobile Application Development Process," *Economy Informatics*, vol. 13, no. 1, p. 40, 2013.

[23] N. Viennot, E. Garcia, and J. Nieh, "A Measurement Study of Google Play," *Performance Evaluation Review*, vol. 42, no. 1, pp. 221–234, 2014.

[24] J. Zbick, I. Nake, M. Jansen, and M. Milrad, "MLearn4web - A Web-based Framework to Design and Deploy Cross-platform Mobile Applications," in *ACM International Conference Proceeding Series*, vol. 2014-Novem, (New York, New York, USA), pp. 252–255, Association for Computing Machinery, 11 2014.

[25] X. Qiao, P. Ren, S. Dustdar, and J. Chen, "A New Era for Web AR with Mobile Edge Computing," *IEEE Internet Computing*, vol. 22, pp. 46–55, 7 2018.

[26] X. Qiao, P. Ren, S. Dustdar, L. Liu, H. Ma, and J. Chen, "Web AR: A Promising Future for Mobile Augmented Reality-State of the Art, Challenges, and Insights," *Proceedings of the IEEE*, vol. 107, no. 4, pp. 651–666, 2019.

[27] J. LI and X. Guo, "Global Deployment Mappings and Challenges of Contact-tracing Apps for COVID-19," *SSRN Electronic Journal*, 5 2020.

[28] E. Laukkanen, J. Itkonen, and C. Lassenius, "Problems, Causes and Solutions when Adopting Continuous Delivery—A Systematic Literature Review," *Information and Software Technology*, vol. 82, pp. 55–79, 2 2017.

[29] B. El Khalyly, A. Belangour, M. Banane, and A. Erraissi, "A New Metamodel Approach of CI/CD Applied to Internet of Things Ecosystem," in *2020 IEEE 2nd International Conference on Electronics, Control, Optimization and Computer Science (ICECOCS)*, pp. 1–6, IEEE, 12 2020.

[30] A. R. Hevner, S. T. March, J. Park, and S. Ram, "Design Science in Information Systems Research," *MIS Quarterly: Management Information Systems*, vol. 28, no. 1, pp. 75–105, 2004.

[31] K. Peffers, T. Tuunanen, M. A. Rothenberger, and S. Chatterjee, "A Design Science Research Methodology For Information Systems Research," *Journal of Management Information Systems*, vol. 24, no. 3, pp. 45–77, 2007.