Technical Disclosure Commons

Defensive Publications Series

November 08, 2018

Dynamic generation of a custom application by combining parts of different applications

Victor Cărbune

Sandro Feuz

Dynamic generation of a custom application by combining parts of different applications

ABSTRACT

Presently, users need to use multiple different applications of the same service provider to access the different functionalities provided by each app, rather than use a single application. As a result, users need to download, install, and maintain a large number of applications on their user devices to utilize various services accessible via their account with the service provider. This disclosure utilizes dynamic assembly of application components to deliver an application package that includes the features of the user's interest from multiple applications of multiple service providers, when service providers allow such combination.

KEYWORDS

- Application delivery
- Application customization
- Dynamic code assembly
- Application parts
- Application components
- Multi-service applications
- Mobile apps

BACKGROUND

Users typically maintain accounts with various service providers that provide different services online, e.g., e-mail, social networking, photo sharing, translation, conference calling etc. Many service providers provide a number of services, e.g., services for messaging, photo sharing, video viewing, conference calling, text translation, etc. Currently, most service providers provide a dedicated application for each of these services. Users do not have options to

combine the various functionalities provided via different applications of the same service provider within a single application. As a result, a user that wishes to use services provided by the service provider needs to download, install, and maintain a corresponding set of applications on their device, e.g., a smartphone, wearable device, tablet, laptop, etc., to utilize the various services accessible via their account with the service provider. Further, users may also utilize services from multiple different providers, each of which provide their own applications.

Moreover, current app configurations require a user to download the entire application on a user device, even when only a part of the application functionality is likely to be used. As a result of the presence of functionality that is unlikely to be used by the user, download of the application uses more download bandwidth, and the application code takes up more storage space on the device and uses more memory during runtime.

DESCRIPTION

Many operating systems (OSs) provide features that allow a larger application to be broken down into smaller pieces based on functionality. Per techniques of this disclosure, the functional pieces are selectively assembled and packaged to generate a customized version of the application. By utilizing such dynamic assembly of application code, this disclosure enables delivery of a single application package that includes only the features of the user's interest from multiple applications. Depending on user preferences and permissions from application developers, the multiple applications from which the single application package is generated may be provided by a single or multiple providers.

The user can specify one or more user accounts with different service providers and associated subset of functionality of interest. For example, the user can indicate that they wish to use video uploading service of one service provider, chat service of another service provider, and

auto rental service of a third service provider. The user's selections are relayed to an application repository, such as an online application store or a website that provides application downloads. The parts of each of the service provider applications that correspond to the desired functionality are assembled and linked together to dynamically generate and deliver a custom application that meets the user's specified functionality and service selections.

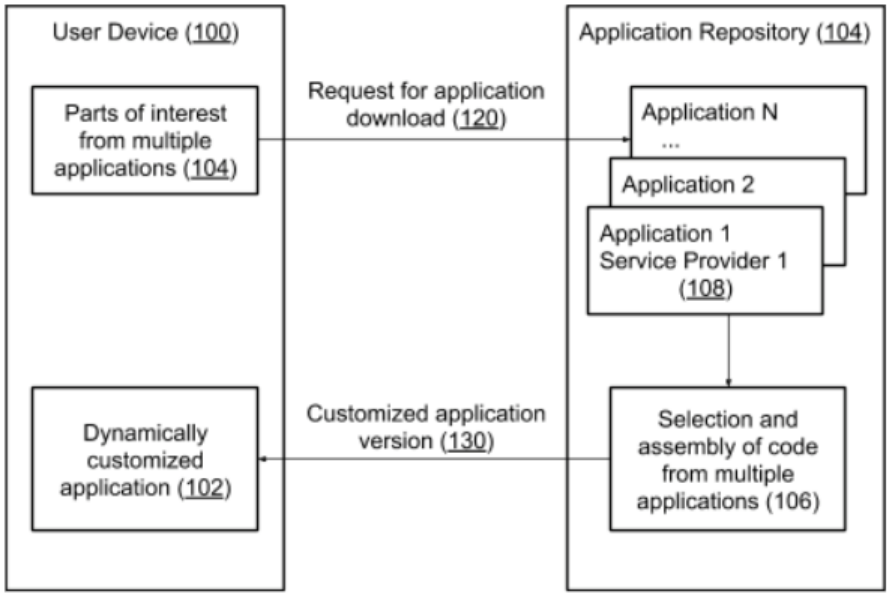

Fig. 1: Customized delivery of a software application dynamically generated by combining parts of several applications

Fig. 1 shows an example implementation of the techniques of this disclosure. A user of a device (100) requests the download (120) of a custom application (102) from an application repository (104), such as an online application store or website that provides application downloads. With user permission, the request includes the user's selection of parts of interest (104) and/or a description of desired functionality for each application. Description and metadata regarding the atomic functional parts of each of the multiple applications stored in the repository are utilized to identify the parts that provide the functionality requested by the user. Application

code (106) that corresponds to the identified parts is assembled to generate and deliver (130) the requested customized application to the user device.

Successful realization of the techniques of this disclosure relies upon modularized applications that are composed of parts that can operate independently of other parts of the application. Each of these parts can be made to operate as a standalone application by removing other parts of the application that are not needed for the operation of the part in question. Aspects of the application, such as state management, that are utilized across multiple parts of the application are included as necessary to ensure that the separated part of the application can operate independently. Modularization is based on, e.g., developer annotations of relevant sections of the application code that indicate the parts of the application or application functionality that is provided by different portions of the code. For example, shared functionality such as user login or authentication is annotated as "necessary" for most parts of the application and is bundled as a dependency with any independent application that is assembled from the main application.

The functionality supported by the specific part extracted from the original application is signaled to other applications and the OS via standard communication protocols between applications and between applications and the OS. When multiple such parts are combined to produce a bundled application, protocols can be layered by priority with the intended operation attempted to be handled within the parts of the bundled application first and being passed down to the OS in case the attempt is unsuccessful. Such handling ensures that a majority of the original functionality of the respective parts of the corresponding applications is supported by the bundled application.

An application composed of a combination of parts from multiple other applications needs to handle the different interfaces utilized by the different originating applications. To address this issue, the various parts extracted from the original applications are placed within a standard separate higher level framework component, such as a main application screen of the combined application with a menu that presents separate options for each of the bundled parts with its own layout and interface. As an alternative, the rendered layouts and the corresponding code can be inspected and readjusted to be rendered more meaningfully within the combined application bundle. The interface readjustments can be based on relevant heuristics or derived from the output of a trained machine learning model. Existing layouts of corresponding application features can be used as the training data for the machine learning model.

An application generated by combining parts of applications from multiple service providers needs seamless interoperation between the providers. In some instances, such interoperation may require linking the user's contacts or data across the various providers accessed by the combined application. The requirement of such linking is indicated to the user, and the user is provided with options to permit or deny permission for linking. When the user provides permission, a contact linking component that executes on the user's device processes the user's contacts from the providers for which permission has been provided to link such contacts across providers. Linking can be performed by matching one or more user permitted factors, e.g. contact name, phone number, etc.

Further, the contact linking is performed and maintained by the underlying framework of the combined application within which the parts of the provider applications are bundled. This allows the user's contacts on a particular service to not be disclosed to other service providers. The contact linking enables a user to select a contact from any of the services for which a

component is present within the combined application and seamlessly communicate with the same contact via any of the other services. For example, consider an application that combines the video sharing functionality of one service and the messaging functionality of another service. When the user indicates that they want to share a link to a video uploaded to the first service via a text message to a contact on the second service, the user can select to share the content with the contact on either of these services. The contact linking enables the combined application to automatically relay the message to the appropriate contact on the messaging service.

The possibility to select features from multiple applications and combine functionality into a single application can provide users with a number of benefits. User experience (UX) is improved when a user can utilize a single custom application that supports multiple tasks. The custom application also eliminates the need for the user to download multiple applications from multiple service providers, thus reducing the bandwidth used for application download and the storage space/ memory utilized to store and execute the task on the user device.

The ability to split an application into independent parts offers application providers the opportunity to explore a variety of delivery and pricing options, such as a-la-carte pricing for different functionalities offered by the application. For example, in addition to the price for the purchase of the entire application, a multimedia application can offer to the user the possibility to buy specific features. For example, a user can buy a set of photo filters from an image editing application, or a feature for stabilization of video recordings from a video editing provider, at prices lower than that of buying the entire application from these providers. Similarly, a user can choose to download only the word processing features instead of downloading an entire office suite application. Application developers can utilize the techniques of this disclosure by incorporating relevant functional parts from other services. For example, developers of a

ridesharing application that supports chatting can embed a chat module from an application of another provider instead of developing custom code for this purpose.

The techniques of this disclosure can be utilized to deliver software applications to any OS or platform that supports splitting applications into components and by any service providers that provide such applications.

Further to the descriptions above, a user may be provided with controls allowing the user to make an election as to both if and when systems, programs or features described herein may enable collection of user information (e.g., information about a user's social network, social actions or activities, profession, a user's preferences, or a user's current location), and if the user is sent content or communications from a server. In addition, certain data may be treated in one or more ways before it is stored or used, so that personally identifiable information is removed. For example, a user's identity may be treated so that no personally identifiable information can be determined for the user, or a user's geographic location may be generalized where location information is obtained (such as to a city, ZIP code, or state level), so that a particular location of a user cannot be determined. Thus, the user may have control over what information is collected about the user, how that information is used, and what information is provided to the user.

CONCLUSION

With permission from users and application developers, this disclosure utilizes dynamic assembly to deliver an application package that includes only the features of the user's interest from multiple applications of multiple service providers. The user can specify one or more service provider accounts and associated subset of functionality of interest. Various parts extracted from the original applications are placed within a standard separate higher level framework component, supporting separate options for each of the bundled parts with its own

layout and interface. The techniques can improve UX, reduce bandwidth consumption for application delivery, and save the storage space/ memory used on the user device to store and execute the application. The techniques also enable service providers to distribute smaller functional components of an application at a lower price.

REFERENCES

1. Google, “About Android App Bundles” available online at https://developer.android.com/guide/app-bundle/, accessed Nov 5, 2018.
2. Wang, Haohong. "A mobile world made of functions" *APSIPA Transactions on Signal and Information Processing* 6 (2017).