Migrating to POLQA 3
POLQA OEM Library

Version 1.1

20 March 2019

1 Contents

2 Preface

This document explains the differences between ITU versions of POLQA, OPTICOM's Library versions and versions of license keys and shows ways to migrate existing code to the latest OEM Library version.

3 Editions, Versions and Generations

In principle we have to distinguish between three different versions, the POLQA ITU *edition*, the OEM Library *version* and the license key *generation*.

- The existing editions of the ITU-T Rec. P.863 are ed. 1.1, 2 (also called 2.4) and 3 (aka POLQAS 3), all marking major milestones in the development of perceptual speech quality measurement.
- The current version of the POLQA OEM Library is 3.2
- Two generations of OPTICOM license keys exist, Gen. 1 and Gen. 2. New keys generated for library version 3.2 and above are all 2nd generation keys. Keys generated until March 2019 where all 1st generation keys.

How these different versions are related is expressed in Table 1 and Table 2 below.

ITU Edition	**Required Key Gen.**	**Supported by Library Versions**
Ed. 1.1	1st or 2nd or hardware dongle	1.6 and later
Ed. 2	1st or 2nd or hardware dongle	1.18 and later
Ed. 3 (POLQA 3)	2nd	3.2 and later

Table 1: Cross reference between ITU editions, library versions and required key generations.

Library Version	**Key Gen.**	**Enabled ITU Edition**
3.2	1st or 2nd or hardware dongle	Ed. 1.1 and ed. 2
3.2	2nd	All (depending on key content)

Table 2: Dependency between key generations and supported POLQA editions of the OPTICOM POLQA OEM Library V3.2.

One way to read Table 1 and Table 2 is to say that the latest OPTICOM POLQA OEM Library (V3.2) implements all editions of POLQA, but the license key determines which POLQA edition is actually supported.

As a consequence, you may update your software to the library version 3.2 without upgrading your license to a POLQA 3 license. In this case you simply continue to use your existing generation 1 key or hardware dongle. The only thing you must take care of is to specify POLQA_V2_4 (or POLQA_V1_1) when calling PolqaLibInit(). See section 6 for more details.

If you want to process according to POLQA 3 (ITU-T P.863 ed. 3), you will need a library version 3.2 (and above) in combination with a 2nd generation key which is valid for POLQA 3. See section 4for more details.

4 Migrating to POLQA 3

In order to migrate your application to POLQA 3, two steps are required,

1. Upgrade your code
2. Upgrade your license (or obtain a new one).

The steps required to upgrade your code are mostly explained in the POLQA OEM Library manual are typically simple. You need to include the new header file, link with the new library version and modify very few functions calls. If license activation and registration are supposed to be part of your application, you may copy the relevant code from the provided demo program. An example code snippet is also listed below.

Upgrading existing 1st generation small volume licenses to 2nd generation will be supported at a later stage. For the moment, in order to take full benefit from POLQA 3, you will need to obtain a new license. License upgrades for large volume customers, are handled through the business model and will require obtaining a new license from the LMS.

```
#include "PolqaDllInterface.h"

#ifdef _WIN32
#ifndef STRING_TYPE
   #define STRING_TYPE    wchar_t
   #define STRING_PREF    L
   #define STRING_COPY    wcscpy
#endif
#else
#ifndef STRING_TYPE
   #define STRING_TYPE    char
   #define STRING_PREF
   #define STRING_COPY    strcpy
#endif
#endif


// LMS account specific
#define DEFAULT_LMS    STRING_PREF""
#define DEVICE_NAME    STRING_PREF"My devicename"
#define PASSWORD       STRING_PREF"MyPW"
#define USERNAME       STRING_PREF"Me"
#define POOLNAME       STRING_PREF"My pool"

// License specific
#define LICENSE_TYPE            STRING_PREF"Evaluation License 7 days"
#define LICENSE_CLASS           STRING_PREF"1"
#define LICENSE_CATEGORY        STRING_PREF"C"
#define LICENSE_POLQA_VERSION   STRING_PREF"3"

// For 1st gen licenses only:
#define LICENSEFILE_NAME  STRING_PREF"full path to the licensefile"

// Demo code which is listed in the manual. We compile it here to ensure correctness.
void RegisterAndActivate()
{
   // LMS account specific
   STRING_TYPE  OEMDeviceID[MAX_STRING_LEN]  = DEVICE_NAME;
   STRING_TYPE  Password[MAX_STRING_LEN]     = PASSWORD;
   STRING_TYPE  UserName[MAX_STRING_LEN]     = USERNAME;
   STRING_TYPE  Pool[MAX_STRING_LEN]         = POOLNAME;

   // License specific
   STRING_TYPE  LicenseType[MAX_STRING_LEN]     = LICENSE_TYPE;
   STRING_TYPE  LicenseClass[MAX_STRING_LEN]    = LICENSE_CLASS;
   STRING_TYPE  LicenseCategory[MAX_STRING_LEN] = LICENSE_CATEGORY;
   STRING_TYPE  Version[MAX_STRING_LEN] = LICENSE_POLQA_VERSION;
   int    EffectiveChannels  = 2;
   bool   UseExisting = false;           // If true, use licenses already existing in the pool,
                                         // otherwise, generate a new license ("self-generated
                                         // license")
```

```
   POLQA_ERRORCODE PolqaErrorcode=POLQA_OK;    // Returncode of Polqa modules
   INSTDATA        InstanceData=0;

   printf("Registering device...\n");
   PolqaErrorcode = PolqaLibRegisterDevice(&InstanceData, UserName, Password, Pool, OEMDeviceID, 0);
   if (POLQA_OK == PolqaErrorcode)
   {
      printf("Activating license...\n");
      PolqaErrorcode = PolqaLibActivateLicense(&InstanceData, LicenseType, LicenseClass, LicenseCategory,
                                               EffectiveChannels, Version, UseExisting, 0);
      if (POLQA_OK == PolqaErrorcode)
      {
         printf("License activation successful\n");
      }
      else printf(" PolqaLibActivateLicense() failed: %d\n", PolqaErrorcode);
   }
   else printf(" PolqaLibRegisterDevice() failed: %d\n", PolqaErrorcode);

   if (PolqaErrorcode != POLQA_OK)
      throw((PolqaErrorcode));
}
```

5 Migrating to the Latest OEM Library Without Migration to POLQA 3

It is possible to use an existing license for POLQA 2.4 or POLQA 1.1or even hardware dongles in combination with the latest OEM Library. It should be recognized however, that that those licenses will not enable execution of POLQA 3, but would be "POLQA 3 ready" in the sense that simply upgrading the license would enable POLQA 3. In this case, you only need to upgrade your application without registering the device or activating a new license from the LMS. As a consequence, those devices will never require internet access, but POLQA features will be limited to those of previous versions of the OEM library.

The code modifications required are very limited. You need to include the new header file, link with the new library version and modify a few function calls (typically by adding "0" parameters…). The demo program contains examples for everything. Make sure to specify POLQA_V2_4 (or POLQA_V1_1) when calling PolqaLiblnit(), since attempts to run POLQA 3 will result in an error! An example code snippet with the most important changes highlighted is listed below.

Furthermore, if you want to continue using hardware dongles, you must call PolqaLibEnableDongleLegacy() before calling PolqaLiblnit().

```
//-----------------------------------------------------------------------------------------------
// Most simmple way to call POLQA...
// pfRefData          Pointer to data vector with reference signal
// pfDegData          Pointer to data vector with degraded signal
// iNrRefSamples      Number of samples in reference signal vector
// iNrDegSamples      Number of samples in degraded signal vector
// ulRefSampleRate;   Sample rate of reference input
// ulDegSampleRate;   Sample rate of degraded input
//
void MostSimplePolqa(  float *pfRefData, int iNrRefSamples, unsigned long ulRefSampleRate,
                       float *pfDegData, int iNrDegSamples, unsigned long  ulDegSampleRate)
{
   INSTDATA      InstanceData=0;             // InstanceData MUST always be initialized to 0!
   POLQA_ERRORCODE  PolqaErrorcode=POLQA_OK; // Returncode of Polqa modules
   POLQA_DLL_HANDLE PolqaHandle=NULL;        // Main handle for Polqa modules. NOTE: this pointer
                                             // has to be initialized to NULL!
   POLQA_DLL_RESULT_DATA *pPolqaRes=NULL;    // Pointer to Polqa results

   unsigned long ulControl = POLQA_RLEVEL_ALL | POLQA_LC_SWIDE_H | POLQA_V2_4;

   STRING_TYPE LicenseFile[_MAX_PATH];
   LicenseFile[0] = 0;                 // This is sufficient for 2nd gen (=POLQA 3) licenses!

   // 1st generation licensefiles only: The full path needs to be specified. For 2nd gen this is not needed!
   STRING_COPY(LicenseFile, LICENSEFILE_NAME);

   // Optional: Enable use of dongles for POLQA 2.4 and earlier.
   // This line can be omitted if no dongles are used.
   PolqaLibEnableDongleLegacy(&InstanceData);

   // Initialize and run POLQA
   PolqaErrorcode = PolqaLibInit(&PolqaHandle, InstanceData, LicenseFile, ulControl, 0);

   if (PolqaErrorcode != POLQA_OK)
      throw((PolqaErrorcode));

   PolqaErrorcode=PolqaLibRun(PolqaHandle, iNrRefSamples, pfRefData, ulRefSampleRate,
                                           iNrDegSamples, pfDegData, ulDegSampleRate);
   if (PolqaErrorcode != POLQA_OK)
      throw((PolqaErrorcode));

   // Get the results
   pPolqaRes=PolqaLibGetResultsPointer(PolqaHandle);

   // Do something with the results here.....

   // Cleanup
   PolqaErrorcode=PolqaLibFree(&PolqaHandle);
}
```

6 Contact Information

OPTICOM

Dipl.-Ing. M. Keyhl GmbH

Nägelsbachstrasse 38

D - 91052 Erlangen

GERMANY

Phone: +49 (0) 91 31 - 5 30 20 - 0

Fax: +49 (0) 91 31 - 5 30 20 - 20

EMail: info@opticom.de

Webseite: http://www.opticom.de

Further information:

http://www.polqa.info

http://www.opticom.de