электронный журнал
МОЛОДЕЖНЫЙ НАУЧНО-ТЕХНИЧЕСКИЙ ВЕСТНИК
Издатель ФГБОУ ВПО "МГТУ им. Н.Э. Баумана". Эл No. ФС77-51038.

УДК 004.9

Сокрытие и восстановление проектной документации в образах QR-кодов

***Круглая Е.И.**, студент*
Россия, 105005, г. Москва, МГТУ им. Н.Э. Баумана,
*кафедра «Системы автоматизи*рованного проектирования»

Научный руководитель: Волосатова Т.М., к.т.н., доцент
Россия, 105005, г. Москва, МГТУ им. Н.Э. Баумана
norenkov@rk6.bmstu.ru

Введение

Данная работа посвящена исследованию QR-кода как способа сокрытия данных и эффективности его использования при кодировании проектной документации. Такой своеобразный способ хранения информации удобнее, чем привычный штрих-код, так как позволяет зашифровать гораздо больший объем информации. QR-код был выбран в качестве объекта исследования из-за своей актуальности и активной применимости во многих сферах жизни человека, таких как СМИ, Интернет, архитектура и др.

1 Постановка задачи

Целью выполнения работы является разработка приложения, позволяющего кодировать вводимые пользователем данные в образах QR-кода.

В состав ПО должны входить следующие элементы:

- ПО пользовательского интерфейса, совместимое с ПО пользовательского интерфейса автоматизированного рабочего места (АРМ);

- программное средство построения QR-кода для кодирования сообщения.

Приложение должно обеспечивать решение следующих задач:

- построение изображения QR-кода сообщения, выбранного пользователем из буфера обмена АРМ: в стандартном формате, либо по выбору пользователя;

- просмотр изображения;

- занесение полученного изображение в буфер обмена АРМ;

- выработка диагностических сообщений при нештатных действиях пользователя.

Исходные данные и результаты обработки данных должны храниться в буфере обмена АРМ. Протокол обмена должен быть согласован с приложением, позволяющем восстанавливать сообщения из QR-кода.

2 QR-код как способ сокрытия данных

QR-код - двумерный штрихкод, разработанный японской фирмой *Denso-Wave*.

В этом штрихкоде кодируется разнообразная информация, состоящая из символов (включая кириллицу, цифры и спецсимволы). Информация может быть любая: адрес сайта, телефон, электронная визитка, координаты местоположения и так далее. Один QR-код (рис 1.) может содержать 7089 цифр или 4296 букв [2, 3].

Рис 1. Пример QR-кода со встроенным сообщением

Как и в случае с товарными штрихкодами, известными как *bar-code*, кодирование информации в определенных графических символах позволяет удобно и быстро считывать эту информацию с помощью специальных сканеров. QR-код, как правило, считывается приложением, установленным на мобильный телефон, после чего устройство действует в зависимости от вида информации, заложенной в QR-код: если это адрес сайта - открывает сайт в браузере, если это электронная визитка - добавляет нового абонента в список контактов, если это обычный текст - просто выводит его на экран.

Таким образом QR-код позволяет автоматически считывать различные данные, а также кодировать большое количество информации в компактном изображении (4296 символов – это более двух машинописных страниц текста) [3].

3 Генерация QR-кода

Процесс генерации QR-кода [1] состоит из нескольких этапов.

3.1. Кодирование данных

Существует четыре основных кодировки QR-кода, в зависимости от используемых символов: цифровая (10 бит на три цифры, до 7089 цифр), буквенно-цифровая (поддерживаются цифры от 0 до 9, буквы от A до Z и спецсимволы $%*+-./: и пробел; 11 бит на два символа, до 4296 символов), байтовая (по умолчанию кодировка *ISO 8859-1)*, до 2953 байт), кандзи (китайско-японские иероглифы; 13 бит на иероглиф, до 1817 иероглифов).

На первом этапе генерации QR-кода выбирается уровень избыточности (коррекции) и версия. Чем выше уровень коррекции, тем более допустимо повреждение QR-кода и тем менее информации можно поместить на QR-коде фиксированного размера. Уровни избыточности делятся в зависимости от процентного соотношения повреждение на 4 типа: L (7%), M(15%), Q(25%), H (30%). От выбранной версии напрямую зависит количество информации в битах, которые можно закодировать.

3.2. Добавление служебной информации. Заполнение

В зависимости от типа кодирования на первом этапе была получена необходимая последовательности бит. Перед последовательностью бит нужно добавить в начале два поля: способ кодирования и количество данных. Способ кодирования — поле длиной 4 бита, которое имеет следующие значения: 0001 для цифрового кодирования, 0010 для буквенно-цифрового и 0100 для побайтового. Количество данных — это количество кодируемых символов, а для побайтового количество байт, представленное в виде двоичного числа, длина которого определяется по таблице 1.

Таблица 1

Длина поля количества данных

	Версия с 1 по 9	Версия с 10 по 26	Версия с 27 по 40
Цифровое	10 бит	12 бит	14 бит
Буквенно-цифровое	9 бит	11 бит	13 бит
Побайтовое	8 бит	16 бит	16 бит

Последовательность бит данных дополняется нулями так, чтобы её длина стала кратна восьми. Затем последовательность разбивается на группы по восемь бит и представляется в виде последовательности байт. Если количество бит в текущей

последовательности байт меньше того, которое нужно для выбранной версии, то её надо дополнить чередующимися байтами 11101100 и 00010001.

3.3. Разделение информации на блоки

Полученная на предыдущем этапе последовательность байтов подвергается разбиению на блоки в зависимости от выбранной версии и уровня коррекции. Если количество блоков равно одному, то можно переходить к другому этапу.

При разделении информации может получиться нецелое число блоков. В таком случае остаток от деления определяет, сколько блоков являются дополненными – блоки, в которых количество байтов больше на один, чем в остальных.

Блок заполняется байтами из данных полностью. Когда текущий блок полностью заполняется, очередь переходит к следующему. Байтов данных должно хватить ровно на все блоки.

3.4. Создание байтов коррекции

Для исправления ошибок применяется алгоритм, основанный на коде Рида-Соломона с 8-битным кодовым словом. В зависимости от уровня избыточности и версии определяется количество байтов коррекции, которые определяют генерирующий многочлен. Алгоритм Рида-Соломона использует многочлен с коэффициентами, определяемыми по таблице 2.

Таблица 2

Генерирующий многочлен

Количество байтов коррекции	Генерирующий многочлен
7	87, 229, 146, 149, 238, 102, 21
10	251, 67, 46, 61, 118, 70, 64, 94, 32, 45
13	74, 152, 176, 100, 86, 100, 106, 104, 130, 218, 206, 140, 78
15	8, 183, 61, 91, 202, 37, 51, 58, 58, 237, 140, 124, 5, 99, 105
16	120, 104, 107, 109, 102, 161, 76, 3, 91, 191, 147, 169, 182, 194, 225, 120
17	43, 139, 206, 78, 43, 239, 123, 206, 214, 147, 24, 99, 150, 39, 243, 163, 136
18	215, 234, 158, 94, 184, 97, 118, 170, 79, 187, 152, 148, 252, 179, 5, 98, 96, 153
20	17, 60, 79, 50, 61, 163, 26, 187, 202, 180, 221, 225, 83, 239, 156, 164, 212, 212, 188, 190
22	210, 171, 247, 242, 93, 230, 14, 109, 221, 53, 200, 74, 8, 172, 98, 80, 219, 134,

	160, 105, 165, 231
24	229, 121, 135, 48, 211, 117, 251, 126, 159, 180, 169, 152, 192, 226, 228, 218, 111, 0, 117, 232, 87, 96, 227, 21
26	173, 125, 158, 2, 103, 182, 118, 17, 145, 201, 111, 28, 165, 53, 161, 21, 245, 142, 13, 102, 48, 227, 153, 145, 218, 70
28	168, 223, 200, 104, 224, 234, 108, 180, 110, 190, 195, 147, 205, 27, 232, 201, 21, 43, 245, 87, 42, 195, 212, 119, 242, 37, 9, 123
30	41, 173, 145, 152, 216, 31, 179, 182, 50, 48, 110, 86, 239, 96, 222, 125, 42, 173, 226, 193, 224, 130, 156, 37, 251, 216, 238, 40, 192, 180

3.5. Объединение блоков

Полученные на предыдущем этапе блоки данных и блоки байтов коррекции объединяются следующим образом: из каждого блока данных по очереди берётся один байт информации, когда очередь доходит до последнего блока, из него берётся байт и очередь переходит к первому блоку. Так продолжается до тех пор, пока в каждом блоке не кончатся байты. Если в текущем блоке уже нет байт, то он пропускается (такое происходит, когда обычные блоки уже пусты, а в дополненных ещё есть по одному байту). Аналогичным образом надо сделать с блоками байтов коррекции. Они берутся в том же порядке, что и соответствующие блоки данных.

В итоге получается следующая последовательность данных:

3.6. Размещение информации на QR-коде

Полученную на предыдущем этапе последовательность байт можно поместить в образ QR-кода (рис. 2).

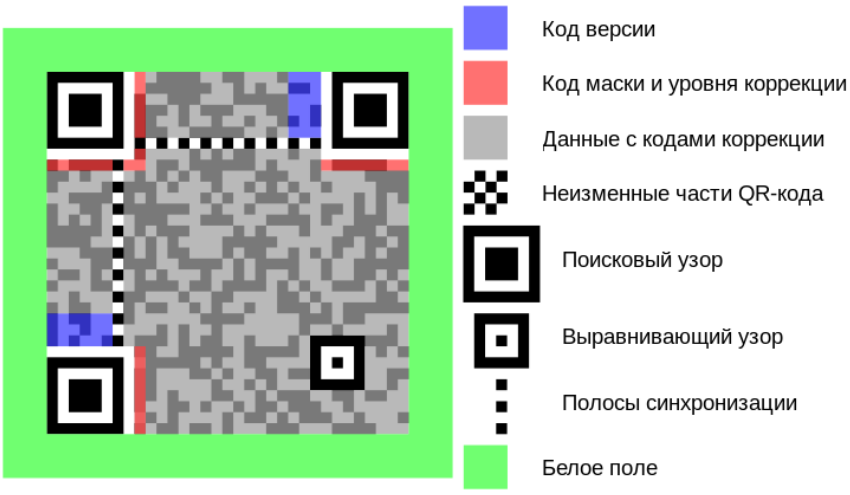

Рис 2. Поля QR-кода

4 Описание программной реализации

Программа создана на языке C++ в среде Qt Creator (рис. 3). Она реализует представленный выше алгоритм построения QR-кода.

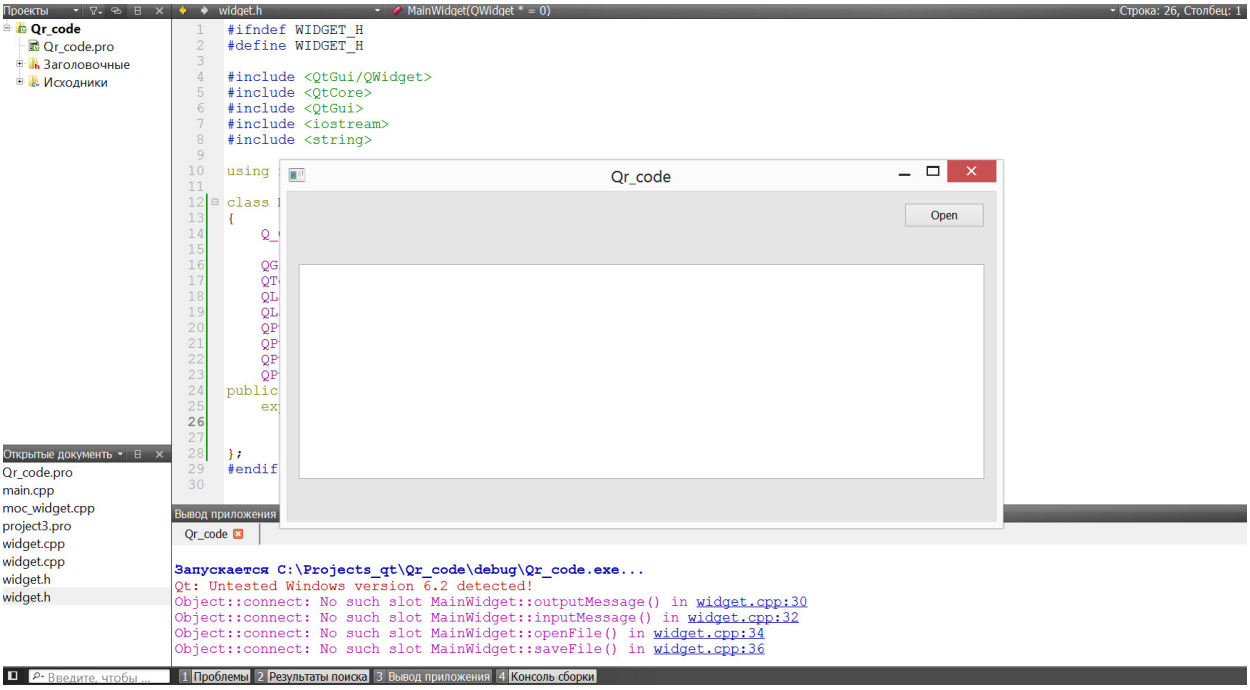

Рис 3. Графический интерфейс приложения

При запуске проекта появляется текстовое окно, дающее возможность ввести требуемое сообщение для кодировки. При нажатии *Write* введенное в окне сообщение подвергается обработке: создается последовательность бит и по заложенному в программе алгоритму выстраивается QR-код. При нажатии *Open* пользователь будет перенаправлен в заданную в программе директорию на компьютере для просмотра QR-кода.

Существенным недостатком программной реализации является то, что пользователь не имеет возможности непосредственного просмотра QR-кода в графическом окне проекта.

Заключение

В рамках работы создано программное средство по сокрытию проектной документации в образах QR-кода. Тестирование приложения показало, что самым оптимальным уровнем избыточности при кодировании сообщений является уровень коррекции M, допускающий 15% повреждений. Устранение недостатков программной

реализации и разработка модуля, позволяющего восстановить сообщение из QR-кода, были определены в качестве направлений для дальнейшей работы.

Список литературы

1. Jason Brown «QR Code Demystified». Series of Articles.
2. QR-код [HTML] (http://qr.astu.org/) (дата обращения: 16.03.2013).
3. Что такое QR-код? [HTML] (http://www.exler.ru/likbez/18-02-2011.htm) (дата обращения: 20.05.2013).