СОЗДАНИЕ БИБЛИОТЕК ИГРОВОГО ПРИЛОЖЕНИЯ «СТАРГЕЙМЕС-ПРО»

Алексеев В.Ф.
Кандидат технических наук, доцент кафедры проектирования информационно-компьютерных систем, Белорусский государственный университет информатики и радиоэлектроники (БГУИР), г. Минск
Мамедов К.Т.
Магистрант кафедры проектирования информационно-компьютерных систем, Белорусский государственный университет информатики и радиоэлектроники (БГУИР), г. Минск

CREATION OF LIBRARIES OF THE GAME APPLICATION «STARGEIMES-PRO»

Alexeev V.
PhD, Associate Professor of Information Systems Design Department, Belarusian State University of Informatics and Radioelectronics (BSUIR), Minsk
Mamedov K.
Master student of the department of information and computer systems design of Belarusian state university of informatics and radioelectronics, Minsk

Аннотация
Рассмотрены аспекты создания и использования библиотек программирования; проведен анализ текущего технологического процесса создания библиотек игрового приложения «СТАРГЕЙМ-ПРО»; разработаны направления совершенствования процесса создания и использования библиотек игрового приложения «СТАРГЕЙМЕС-ПРО».

Abstract
Aspects of creation and use of programming libraries are considered; analysis of the current technological process of creating libraries for the game application "STARGAME-PRO"; directions of improving the process of creating and using libraries of the game application "STARGEIMES-PRO" have been developed.

Ключевые слова: компьютерные игры, игровые приложения, библиотеки игровых приложений, Android.

Keywords: computer games, game applications, game application libraries, Android.

Введение. Компьютерные игры представляют собой явление информационного общества, приобретшее в последние десятилетия исключительную популярность. Начиная с появления первых компьютерных игр, происходящих в виртуальном пространстве, миновало около тридцати лет. Несмотря на своё широкое распространение и социокультурную институционализацию, они ещё не получили научную и философскую разработку, анализ и определение места в системе информационной культуры и социальных структур.

Компьютерные игры и игровые приложения имеют различные направления использования, одним из которых является развитие профессиональных навыков различных направлений посредством метода «геймификации».

Несмотря на некоторый скепсис части общества в отношении использования подхода «геймификации» и пользы игр для взрослых, они продолжают выполнять образовательную функцию. Постепенно они становятся частью обучающих программ для множества профессий.

Важным аспектом в рамках создания игровых приложений является степень оптимизации процесса разработки. Одним из способов, упрощающих деятельность разработчиков, является созда-

ние библиотек программирования, способных автоматизировать отдельные этапы разработки. Библиотека в программировании представляет собой сборник подпрограмм или объектов, используемых для разработки программ. Однако в практике деятельности ИТ-компаний применение библиотек программирования сопряжено с рядом сложностей, мешающих процессу разработки, приводящих к увеличению трудовых и временных затрат и, соответственно, удорожанию стоимости конечного продукта.

Актуальность исследований. Актуальность темы исследования сопряжена с необходимостью оптимизации процесса разработки исследуемого игрового приложения «СТАРГЕЙМЕС-ПРО», а также иных приложений компании. Это позволит не только оптимизировать основной бизнес-процесс, связанный с разработкой данного приложения, потенциально повысив тем самым его конкурентоспособность на рынке, но также будет облегчать работу над дальнейшими проектами, прямо или косвенно связанными с игровым приложением «СТАРГЕЙМЕС-ПРО».

Проблема оптимизации разработки приложений посредством использования библиотек рассматривается в большей степени на тематических профессиональных ресурсах, в академической среде данный вопрос, к сожалегнию, не имеет значительной разработки.

Тем не менее, ряд аспектов отражен в исследованиях следующих авторов: А. Посконина (проблемы масштабируемости), А. П. Кондратюк (библиотеки как инструменты оптимизации разработки приложений для мобильных операционных систем), Р. Мартина (вопросы создания «чистого» кода) и др. [1-6].

Характеристика используемых библиотек игровых приложений. В процессе разработки игрового приложения «СТАРГЕЙМЕС-ПРО» используется кроссплатформенный «фреймворк» libGDX. Он основан на языке программирования Java с некоторыми компонентами, написанными на C и C++ для повышения производительности определенного кода.

«Фреймворк» libGDX поддерживает Windows, Linux, Mac OS X, Android, iOS и HTML5 как целевые платформы. Распространение LibGDX осуществляется в соответствии с лицензией Apache 2.0 (лицензия на свободное программное обеспечение Apache Software Foundation), что дает право использовать libGDX для любых целей, свободно распространять, изменять, и распространять изменённую копию, за исключением названия.

libGDX позволяет разработчику писать, тестировать и отлаживать код на собственном компьютере и затем переносить его на другие ОС. Для этого «фреймворк» использует отдельные модули для сборки приложения под каждую платформу, а так же независимый модуль, который содержит основной код приложения. libGDX позволяет написать код однажды и затем развертывать игру или приложение на нескольких платформах без модификации. Так же можно разрабатывать приложение на основном компьютере и получать огромную выгоду быстрой разработки вместо того, чтобы ждать, когда последние изменения буду внедрены и установлены на устройство и будут скомпилированы в HTML5. Можно использовать все инструменты Java, чтобы быть продуктивным, насколько это возможно. libGDX позволяет перейти на любой низкий уровень, давая примой доступ к файловой системе, устройствам ввода, аудио устройствам и OpenGL через единый OpenGL ES 2.0 и 3.0 интерфейс.

Наверху таких низкоуровневых возможностей создан мощный набор API, который позволяет решать общие в разработке игр задачи, такие как визуализация спрайтов, теста, построение пользовательских интерфейсов, проигрывание звуковых эффектов и музыки, линейная алгебра и тригонометрические вычисления, разбор JSON и XML, и так далее.

При необходимости, libGDX может перейти от Java к нативному коду, чтобы сосредоточиться на самой лучшей и возможной производительности. Весь этот функционал скрывается за Java API, так что разработчик не должен беспокоится о кросс-компилировании нативного кода на всех платформах. Многие части libGDX знают специфику платформы и разработчику не нужно с ними сталкиваться [7]:

libGDX сосредоточен на том, чтобы являться фрейворком, нежели движком, признавая, что нет универсального решения. libGDX предоставляет мощные абстракции, которые позволяют выбирать, как создавать игру или приложение;

libGDX для своей функциональности использует множество сторонних библиотек, представленных в соответствии с таблицей 1.

Таблица 1

Характеристика используемых сторонних библиотек в «СТАРГЕЙМЕС-ПРО»

Наименование сторонней библиотеки	Характеристика
Lightweight Java Game Library (LWJGL)	Библиотека Java, которая включает межплатформенный доступ к популярным встроенным API, полезным в разработке графики (OpenGL), аудио (OpenAL) и параллельные вычисления (OpenCL) приложения. Этот доступ прямой и высокоэффективный, все же также обернутый в безопасный с точки зрения типов и удобный для пользователя уровень, подходящий для экосистемы Java
OpenGL	Главная среда для разработки переносимых, интерактивных 2D и 3D графических приложений. Начиная с его введения в 1992 г., OpenGL стал отраслью, наиболее широко используемой, и поддерживающей 2D и 3D графические прикладные программные интерфейсы (API), принося тысячи приложений к большому разнообразию компьютерных платформ. OpenGL способствует инновациям и разработке приложений скоростей, включая широкий набор рендеринга, отображения текстур, специальных эффектов и других мощных функций визуализации
FreeType	Библиотека программного обеспечения в свободном доступе, для визуализации шрифтов
OpenAL	Межплатформенный 3D аудио API, подходящий для использования с игровыми приложениями и многими другими типами аудио приложений. Библиотека моделирует набор источников звука, перемещающихся в 3D пространство, которые слышит единственный слушатель в пространстве. Основные объекты OpenAL – слушатель, источник и буфер. Может быть большое количество буферов, которые содержат аудиоданные. Каждый буфер может быть присоединен к одному или более источникам, которые представляют точки в 3D пространстве, которые испускают аудио.
SoundTouch Audio Processing Library	Библиотека обработки аудиоданных с открытым исходным кодом для изменения темпа, подачи и скоростей воспроизведения аудиопотоков или аудиофайлов. Библиотека дополнительно поддерживает оценку стабильных уровней ударов в минуту для аудиотреков.
Vorbis	Абсолютно открытая, профессиональная технология аудиокодирования и потоковой передачи без патентов со всеми преимуществами открытого исходного кода.
mpg123	Библиотека содержащая работающий в реальном времени MPEG 1.0/2.0/2.5 аудиоплеер / декодер для уровней 1,2 и 3 (обычно уровень 3 MPEG 1.0 иначе MP3), а также допускающие повторное использование библиотеки декодирования и вывода.
Box2D	Библиотека написанная на C++, с открытым исходным кодом для моделирования твердых тел в 2D. Box2D разработан Erin Catto и имеет zlib лицензию. zlib лицензия не требует подтверждения, что позволяет свободно использовать Box2D в своем продукте.
Kiss FFT («Keep It Simple, Stupid.» Fast Fourier Transform)	Очень небольшая, довольно эффективная, смешанная библиотека FFT основания, которая может использовать или фиксированную точку или типы данных с плавающей точкой

Примечание – Источник: [8-16]

Роль библиотек игрового приложения «СТАРГЕЙМЕС-ПРО» в общей архитектуре. Формирование архитектуры игрового приложения «СТАРГЕЙМЕС-ПРО» осуществляется из набора компонентов. Каждый компонент построен на основе элементов более низкого уровня. Так, архитектура Android с размещением библиотеками игрового приложения представлена в соответствии с рисунком 2.

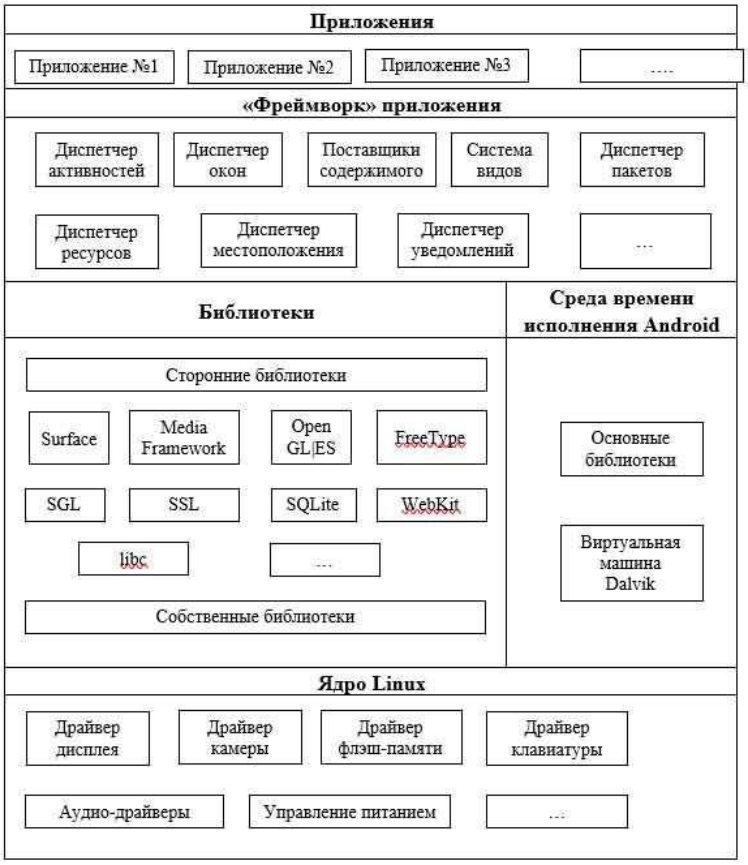

Рисунок 2 – Архитектура Android с библиотеками игрового приложения

В нижней части рисунка 2.1 можно отметить, что ядро предлагает основные драйверы для аппаратных компонентов системы. Кроме того, ядро отвечает за память, управление процессами, поддержку сети и т.д. Среда выполнения Android, являющаяся надстройкой над ядром, отвечает за порождение и выполнение приложения Android. Каждая программа работает в собственном процессе со своей виртуальной машиной Dalvik.

Алгоритм создания библиотек игрового приложения «СТАРГЕЙМЕС-ПРО». В рамках разработки игровых приложений «СТАРГЕЙМЕС-ПРО» используется кроссплатформенный «фреймворк» libGDX. Он основан на языке программирования Java с некоторыми компонентами, написанными на C и C++ для повышения производительности определенного кода.

Помимо этого, разработчиками осуществляется самостоятельное создание библиотек.

Алгоритм создания статистической библиотеки. Статическая библиотека создается из обычных объектных файлов путем их архивации с помощью утилиты ar.

Все действия, которые описаны ниже выполняются в каталоге library (т.е. туда надо перейти командой cd). Просмотр содержимого каталога выполняется с помощью команды ls или ls -l.

Тогда будут получены объектные файлы:

gcc -c ./source/*.c

В итоге в каталоге library наблюдается следующее:

figures.o mylib.h source text.o

Далее используется утилиту ar для создания статической библиотеки:

ar r libmy1.a *.o

Параметр r позволяет вставить файлы в архив, если архива нет, то он создается. Далее указывается имя архива, после чего перечисляются файлы, из которых архив создается. Если объектные файлы не требуются, их можно удалить:

rm *.o

В итоге содержимое каталога library выглядит следующим образом:

libmy1.a mylib.h source,

где libmy1.a – это статическая библиотека.

Алгоритм создания динамической библиотеки. Объектные файлы для динамической библиотеки компилируются особым образом. Они должны содержать так называемый позиционно-независимый код (англ. position independent code). Наличие такого кода позволяет библиотеке подключаться к программе, когда последняя загружается в память. Это связано с тем, что библиотека и программа не являются единой программой, а значит как угодно могут располагаться в памяти относительно друг друга. Компиляция объектных файлов для динамической библиотеки выполняется с опцией -fPIC компилятора gcc:

gcc -c -fPIC source/*.c

В отличие от статической библиотеки динамическую создают при помощи gcc указав опцию -shared:

gcc -shared -o libmy2.so *.o

Использованные объектные файлы можно удалить:

rm *.o

В итоге содержимое каталога library:

libmy1.a libmy2.so mylib.h source

Использование библиотеки в программе. Далее, когда в каталоге project присутствуют два объектных файла: main.o и data.o, они должны быть скомпилированы в исполняемый файл project посредством объединения со статической библиотекой libmy1.a. Это осуществляется с помощью следующие команды:

gcc -o project *.o -L../library -lmy1

Начало команды должно быть понятно: опция -o указывает на то, что компилируется исполняемый файл project из объектных файлов.

Помимо объектных файлов проекта в компиляции участвует и библиотека. Об этом свидетельствует вторая часть команды: -L../library -lmy1. Здесь опция -L указывает на адрес каталога, где находится библиотека, он и следует сразу за ней. После опции -l записывается имя библиотеки, при этом префикс lib и суффикс (неважно .a или .so) усекаются. Необходимо обратить внимание на то, что после данных опций пробел не ставится.

Опцию -L можно не указывать, если библиотека располагается в стандартных для данной системы каталогах для библиотек. Например, в GNU/Linux это /lib/, /urs/lib/ и др.

Запустив исполняемый файл project и выполнив программу, можно увидеть на экране то, что представлено в соответствии с рисунком 3.

Рисунок 3 – Пример результата (условный пример, аналогичный практике компании)

Далее необходимо определить размер файла project:

pl@desk:~/c/project$ ls -l project

-rwxr-xr-x 1 pl pl 8648 ноя 19 07:46 project

Его размер равен 8698 байт.

Компиляция проекта с динамической библиотекой. Перед компиляцией необходимо удалить исполняемый файл и получить его уже связанным с динамической библиотекой. Команда компиляции с динамической библиотекой выглядит следующим образом (одна команда разбивается на две строки с помощью обратного слэша и перехода на новую строку):

gcc -o project *.o \

> -L../library -lmy2 -Wl,-rpath,../library/

Здесь в отличии от команды компиляции со статической библиотеки добавлены опции для линковщика: -Wl,-rpath,../library/. -Wl – это обращение к линковщику, -rpath – опция линковщика, ../library/ – значение опции. Получается, что в команде мы два раза указываем местоположение библиотеки: один раз с опцией -L, а второй раз с опцией -rpath. Видимо для того, чтобы понять, почему так следует делать, потребуется более основательно изучить процесс компиляции и компоновки программ на языке С.

Следует заметить, что если осуществлять компиляцию программы, используя приведенную команду, то исполняемый файл будет запускаться из командной строки только в том случае, если текущий каталог project. При смене каталога, будет возникать ошибка из-за того, что динамическая библиотека не будет найдена. Но если скомпилировать программу следующим образом:

gcc -o project *.o -L../library -lmy2 \

> -Wl,-rpath,/home/pl/c/library

то есть, указать для линковщика абсолютный адрес, то программа в данной системе будет запускаться из любого каталога.

Размер исполняемого файла проекта, связанного с динамической библиотекой, получился равным 8544 байта. Это немного меньше (на 154 байта или на 1,77 %), чем при компиляции проекта со статической библиотекой.

Если посмотреть на размеры библиотек:

pl@desk:~/c/library$ ls -l libmy*

-rw-r--r-- 1 pl pl 3712 ноя 19 07:35 libmy1.a

-rwxr-xr-x 1 pl pl 7896 ноя 19 07:36 libmy2.so

то видно, что динамическая больше статической, хотя исполняемый файл проекта со статической библиотекой больше. Это доказывает, что в исполняемом файле, связанном с динамической библиотекой, присутствует лишь ссылка на нее.

Алгоритм создания общей библиотеки. В качестве одного из наиболее важных аспектов при создании общей библиотеки игрового приложения является сам код, который должен быть помещен в библиотеку (при формировании общей библиотеки кода). В рамках разработки «СТАРГЕЙМЕС-ПРО» при осуществлении попыток формирования данной библиотеки кода размещался только тот код, который был необходим в рамках одного проекта, а далее был не нужен.

В дальнейшем при совершенствовании алгоритма создания общей библиотеки для ее оптимальной работы добавлялся только тот код, который мог быть использован повторно. Данный подход является более рациональным ввиду того, что он значительным образом экономит временные затраты на разработку проекта.

Следует отметить, что в рамках разработки игрового приложения присутствует значительное количество кода специфического характера, который встречается в конкретном игровом приложении. В связи с этим в рамках своей деятельности разработчики «СТАРГЕЙМЕС-ПРО» выносят в общую библиотеку реализации тех игровых приложений, чей «геймплей» с большой долей вероятности будет скопирован в следующих проектах (полностью или частично). В рамках данной деятельности требуется системный подход и выделение в имеющихся решений общих частей.

Таким образом, создание общей библиотеке осуществляется по следующему алгоритму, представленному в соответствии с рисунком 4.

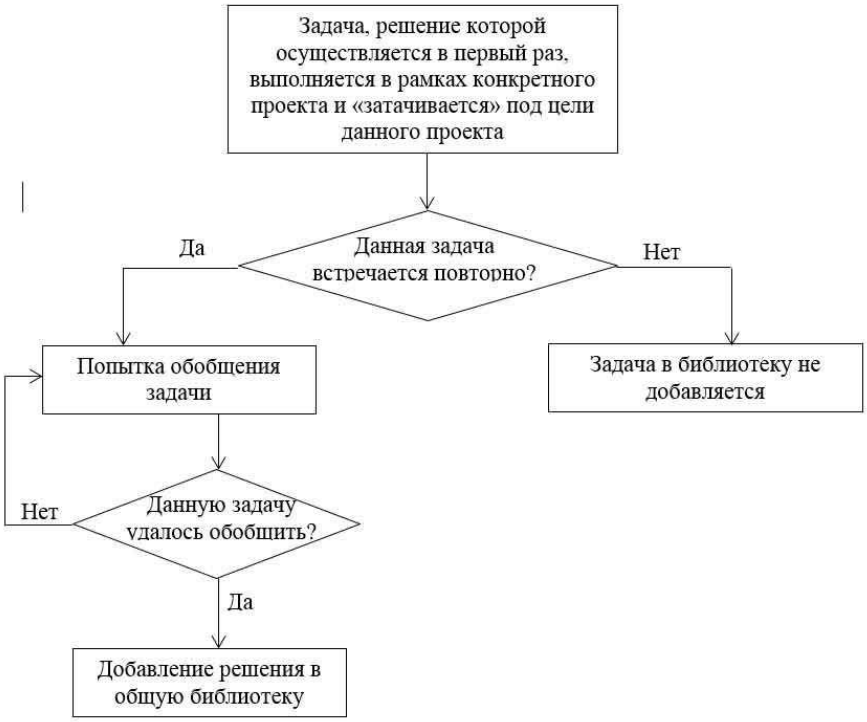

Рисунок 4 – Алгоритм создания общей библиотеки в практике компании «СТАРГЕЙМЕС-ПРО»

Следует отметить, что в практике «СТАРГЕЙМЕС-ПРО» некоторые архитектурные решения проходили несколько итераций изменений (около 5-7) на разных проектах до момента выявления закономерности и вынесения решения в библиотеку.

Заключение. В рамках изучения теоретических аспектов был сделан вывод о том, что навык обращения с библиотеками является одним из обязательных для разработчика.

Библиотеки могут быть разделены на такие виды, как статические и динамические. Статические могут быть представлены в виде исходного текста, подключаемого программистом к своей программе на этапе написания. Динамические библиотеки представляют собой часть основной программы, которая загружается в ОС по запросу работающей программы в ходе ее выполнения, то есть, динамически.

В процессе разработки командой разработчиков при создании приложения могут быть использованы как библиотеки и «фреймворк», так и создано ПО с нуля. Так, например, команда с большим количеством младших разработчиков может работать эффективнее с «фреймворком», который будет обеспечивать определенную структуру и определенные стандарты, помогая команде в создании продукта. С другой стороны, опытной команде, работающей над приложением, которое имеет нишевую функциональность и обладает определенной специфичностью, может быть лучше использовать библиотеку (или библиотеки), так как это может предоставить большую гибкость и контроль.

Список литературы

1. Poskonin A. Web applications and data: achieving abstraction and scalability. Proceedings of the Institute for System Programming of the RAS (Proceedings

of ISP RAS). 2012;23. (In Russ.) https://doi.org/10.15514/ISPRAS-2012-23-10

2. Кондратюк, А.П. Разработка приложений для мобильных операционных систем «Android»: ЭУМК для студ. второй ступени (магистратуры) специальности 1-31 81 06 «Веб-программирование и интернет-технологии» физ.-мат. фак. / А.П. Кондратюк; Брест. гос. ун-т им. А.С. Пушкина, каф. ПМиИ. – Брест: электрон. издание БрГУ, 2016. – 469 с.

3. Мартин Р. Чистый код: создание, анализ и рефакторинг. – СПб: Питер, 2019. – 464 с.

4. Макконнелл С. Совершенный код. Мастер класс / Пер. с англ. – 2-е издание. – СПб.: Питер; М.: Русская редакция, 2010. – 889 с.

5. Физерс Майкл К. Эффективная работа с унаследованным кодом / Физерс Майкл К. – М.: Вильямс, 2009. — 400 с.

6. Кнут Д. Искусство программирования. Том 1. Основные алгоритмы. В 4-х томах. Пер. с англ. – 3-е изд. – М.: Вильямс, 2006. – 682 с.

7. Знакомьтесь, Android Studio [Электронный ресурс] / Meet Android Studio | Android Studio. – 2020. – Режим доступа: https://developer.android.com/studio/intro/index.html #the_user_interface.

8. About [Электронный ресурс]. – 2020. – Режим доступа: http://box2d.org/about.

9. FreeType // The FreeType Project [Электронный ресурс]. – 2020. – Режим доступа: https://www.freetype.org

10. Kiss FFT [Электронный ресурс]/ Kiss FFT download | SourceForge.net. – 2020. – Режим доступа: https://sourceforge.net/projects/kissfft

11. mpg123 – Fast console MPEG Audio Player and decoder library [Электронный ресурс] – 2020. / mpg123, Fast MP3 Player for Linux and UNIX systems. – Режим доступа: http://mpg123.org

12. OpenGL Overview [Электронный ресурс]. – 2020. – Режим доступа: https://www.opengl.org/about

13. SoundTouch Audio Processing Library [Электронный ресурс]. – 2020. – Режим доступа: http://www.surina.net/soundtouch

14. What is LWJGL 3? [Электронный ресурс]/ LWJGL – Lightweight Java Game Library. – 2020. – Режим доступа: https://www.lwjgl.org

15. What is OpenAL? [Электронный ресурс] – 2020. / OpenAL: Cross Platform 3D Audio. – Режим доступа: http://www.openal.org

16. Электронный портал Vorbis.com [Электронный ресурс]. – 2020. – Режим доступа: http://vorbis.com. – Дата обращения: 25.03.2020. [19] *Datasheet EMM5068VU* [Электронный ресурс]. – 2013. – Режим доступа: https://www.sedi.co.jp/pdf/EMM5068VU_ED3-0