УДК 004.42

# ПРОГРАММНОЕ СРЕДСТВО ДЛЯ АВТОМАТИЗАЦИИ ВЕРСТКИ И ПОДГОТОВКИ К ПЕЧАТИ МАТЕРИАЛОВ ДЛЯ НАСТОЛЬНЫХ ИГР

*Желенок Д.А., студент, Савенко А.Г., маг. техн. наук*

*Белорусский государственный университета информатики и радиоэлектроники,*
*Институт информационных технологий,*
*г. Минск, Республика Беларусь*

*Савенко А.Г. маг. техн. наук, ст. препод. каф. ИСиТ*

**Аннотация.** В работе представлено программное средство, предназначенное для автоматизации верстки и подготовки к печати материалов настольных игр. Программное средство имеет графический и консольный интерфейс, продуманную архитектуру, позволяющую не дублировать исходный код алгоритмов для различных интерфейсов. Основными функциями программного средства являются: подготовка файла спецификации с широким выбором параметров (различные виды материалов полиграфической продукции, форматы листов (в том числе рулонная печать), варианты исходных файлов материалов настольных игр и способов их сопоставления, выбор стратегии упаковки); генерация документов для печати в соответствии со спецификацией. Алгоритмы программного средство решают задачи двумерной упаковки и гильотинного раскроя. Применение программного средства позволит сократить затраты на подготовку и производство настольных игр.

**Ключевые слова.** Издание полиграфической продукции, бизнес-процесс, оптимизационные задачи, задача двумерной упаковки, задача гильотинного раскроя, автоматизация процесса, файл спецификации.

***Введение.*** Сегодня настольные игры являются распространенным видом увлечения, образующим целую индустрию. Они имеют большое разнообразие как в оформлении и тематике, так и в жанрах, начиная от абстрактных и логических игр, заканчивая стратегическими и ролевыми играми. Существует много сообществ, клубов, а также интернет-ресурсов, посвященных настольным играм.

Процесс разработки новых игр или дополнений к существующим, чаще всего является итеративным. Это означает что, прежде чем создать финальную версию продукта, разрабатывается прототип, на нем проверяются заложенные идеи и механики через непосредственно, игру, затем вносятся правки, снова проверяются заложенные идеи и т.д. Каждый цикл прототипирования и проверки является отдельной итерацией разработки игры. После разработки готовой версии продукта автор может связаться с издателями, чтобы представить им свои наработки и, возможно, издать через них готовую к продаже игру. Также он может попробовать самостоятельно издать ее, работая напрямую с типографиями. Или как альтернативный вариант, выпустить ее материалы в цифровом виде, как бесплатно, так и платно через специализированные ресурсы.

Кроме того, очередное переиздание тиражей настольных игр нередко сопровождается внесением изменений в правила и механики игры. Изменения в правилах также не редко сопровождаются изменением как информации, содержащейся на игровых компонентах, так и самих компонентов.

***Основная часть.*** Итеративный подход и выпуск или перевыпуск настольной игры в любом виде, подразумевает выполнение верстки компонентов, для дальнейшей их печати. Упрощение и увеличение качества верстки позволяет уменьшать расходы на издание настольных игр. Это, в свою очередь, позволяет снизить стоимость отдельных экземпляров, что повышает доступность, позволяющую охватить большую аудиторию.

Для изготовления различных видов компонентов используются различные материалы. Так, например, для игровых костей чаще всего используется пластмасса, для планшетов и полей плотный картон, а для карт картон или плотная многослойная бумага. Таким образом разные компоненты требуют разной процедуры подготовки для дальнейшего изготовления.

Подготовка компонентов к изготовлению отличается на этапе разработки и издания. Так из-за итеративности процесса разработки, изменения вносятся часто, но направленны они преимущественно на правила и игровые механики, а не на дизайн компонентов (хотя, и он может продумываться и изменяться на данном этапе). На этапе издания дизайн должен приобрести свой конечный вид, который будет удобен как в производстве, так и конечному потребителю.

Для таких компонентов как книги правил, планшеты, поля и карты необходимо предварительно сверстать документ, который можно будет распечатать. Версткой документа называется расположение текста, таблиц, картинок и других элементов на странице документа. В контексте настольных игр процесс верстки подразумевает расположение игровых элементов, так чтобы их было удобнее напечатать и разрезать. При этом формат бумаги и компоновка объектов зависит от того, где в дальнейшем изготавливаются сверстанные объекты. Так на этапе разработки игры логично форматировать карты на листах А4, это позволит легко распечатать их на домашнем принтере или в местном центре полиграфии. Для типографий может больше подойти широкоформатная или рулонная печать.

Разработанное программное средство (ПС) позволяет автоматизировать процесс верстки и подготовки материалов настольных игр к печати. ПС поддерживает гибридный режим работы, что означает поддержку как графического, так и текстового (консольного) варианта интерфейса. Графический режим работы предоставляет пользователю удобный интерфейс для создания файла спецификации и его применение для генерации документов. Файл спецификации – это файл в формате YAML который в декларативной форме описывает параметры, которые ожидаются от сформированных документов. YAML – это удобный для пользователя язык сериализации данных для всех языков программирования. Расшифровка YAML является рекурсивным акронимом и расшифровывается как «YAML Ain't Markup Language» («YAML - не язык разметки») [1]. Использование данного формата предоставляет как простую обработку со стороны программы, так и возможность вручную отредактировать файл без необходимости использовать графический интерфейс.

Файл спецификации описывает следующие параметры:

— структуру папок, в которые будут сохранены результат работы программы;

— количество и название документов, которые пользователь ожидает получить;

— формат бумаги и отступы, которые должны быть соблюдены в каждом из документов;

— источник изображений игровых компонентов и их ожидаемый размер;

— стратегия упаковки объектов на каждой странице;

— источник изображений для обложек, если компоненту они требуются;

— способ формирования меток для реза, если они необходимы.

Каждый из пунктов приведенных выше позволяет настроить ожидаемый результат. Программа, считывая спецификацию и остальные необходимые файлы, может предоставить следующие функции:

— расстановка изображений на двумерной ограниченной области с учетом отступов, ожидаемых размеров компонентов и выбранной пользователем стратегии упаковки;

— масштабирование входных изображений, а также формирование рамки вокруг них в том случае, если необходимо избежать проблем, связанных со смещением бумаги во время печати;

— нарезка входных изображений, в случае если они не влезают в ожидаемую область;

— расстановка меток для реза;

— генерация и расстановка рубашек для тех компонентов, которым она необходима;

— возможность распределить компоненты как в рамках одного документа, так и в разных.

На выходе программа формирует структуру из папок, в которых находятся сгенерированные файлы в формате Portable Document Format (PDF). Структура папок также описывается в исходной спецификации.

Упаковка – класс задач комбинаторной геометрии, включающих в себя размещение фигур заданного размера или формы внутри другой заданной фигуры с наибольшей экономией или с некоторыми другими ограничениями [2]. Под стратегией упаковки подразумевается алгоритм, по которому будет осуществляться расстановка игровых компонентов на страницах документов. В разработанном программном средстве алгоритмы упаковки подразделяются на две основные группы:

– оптимизирующие используемое пространство;

– упрощающие послепечатную обработку.

Первые представляют из себя алгоритмы, решающие задачу двумерной упаковки (2-Dimensional Bin Packing, 2DBP). Они должны быть способны работать с компонентами разных размеров и их результат должен, по возможности, минимизировать занимаемое на листах пространство и количество используемых листов бумаги.

Вторые должны минимизировать количество гильотинных разрезов, которые необходимо совершить чтобы вырезать все компоненты. Эта проблема также известна как задача гильотинного раскроя (Guillotine cutting).

Задача, решаемая алгоритмами двумерной упаковки, формулируется следующим образом: даны прямоугольники определенного размера (задана ширина, высота), которые необходимо расположить на прямоугольниках большего размера (контейнеры). Основной целью является нахождение такого расположения всех прямоугольников, чтобы они не пересекались и занимали наименьшее количество контейнеров [3]. У задачи двумерной упаковки есть два варианта входных данных: когда набор упаковываемых объектов известен заранее (offline-проблема) и когда данные поступают порциями (online-проблема). В разработанном программном средстве применяются алгоритмы, решающие offline-проблему, так как размер всех игровых компонентов известен заранее. Еще одной вариацией задачи двумерной упаковки, которая алгоритмически реализована в данном программном средстве, является задача двумерной упаковки в двумерную полуограниченную полосу (2-Dimensional Strip Packing, 2DSP). Данный вариант отличается от 2DBP тем что упаковка осуществляется только в один контейнер, который ограничен по ширине, но не ограничен по высоте. Вместо минимизации количества контейнеров, мы минимизируем высоту заполненности одного [4].

Задачи 2DBP и 2DSP относятся к классу NP-трудных задач [3, 4]. Из-за этого исследования сфокусированы главным образом на разработке приближенных алгоритмов решения. Приближенные алгоритмы находят оптимальное решение с определенной точностью, но не гарантируют

оптимальной упаковки для любого набора данных. При этом критерий оптимальности зависит от того, что должно быть оптимизированно.

Задача гильотинного раскроя – задача комбинаторной геометрии, близкая к задаче раскроя и задачам упаковки в контейнеры. Основной целью является получение максимального числа листов прямоугольного размера из листа большего размера, делая только гильотинные разрезы, то есть прямые разрезы от края до края. Задача гильотинного раскроя также относится к классу NP-трудных задач [5].

Диаграмма вариантов использования разработанного программного средства представлена на рисунке 1.

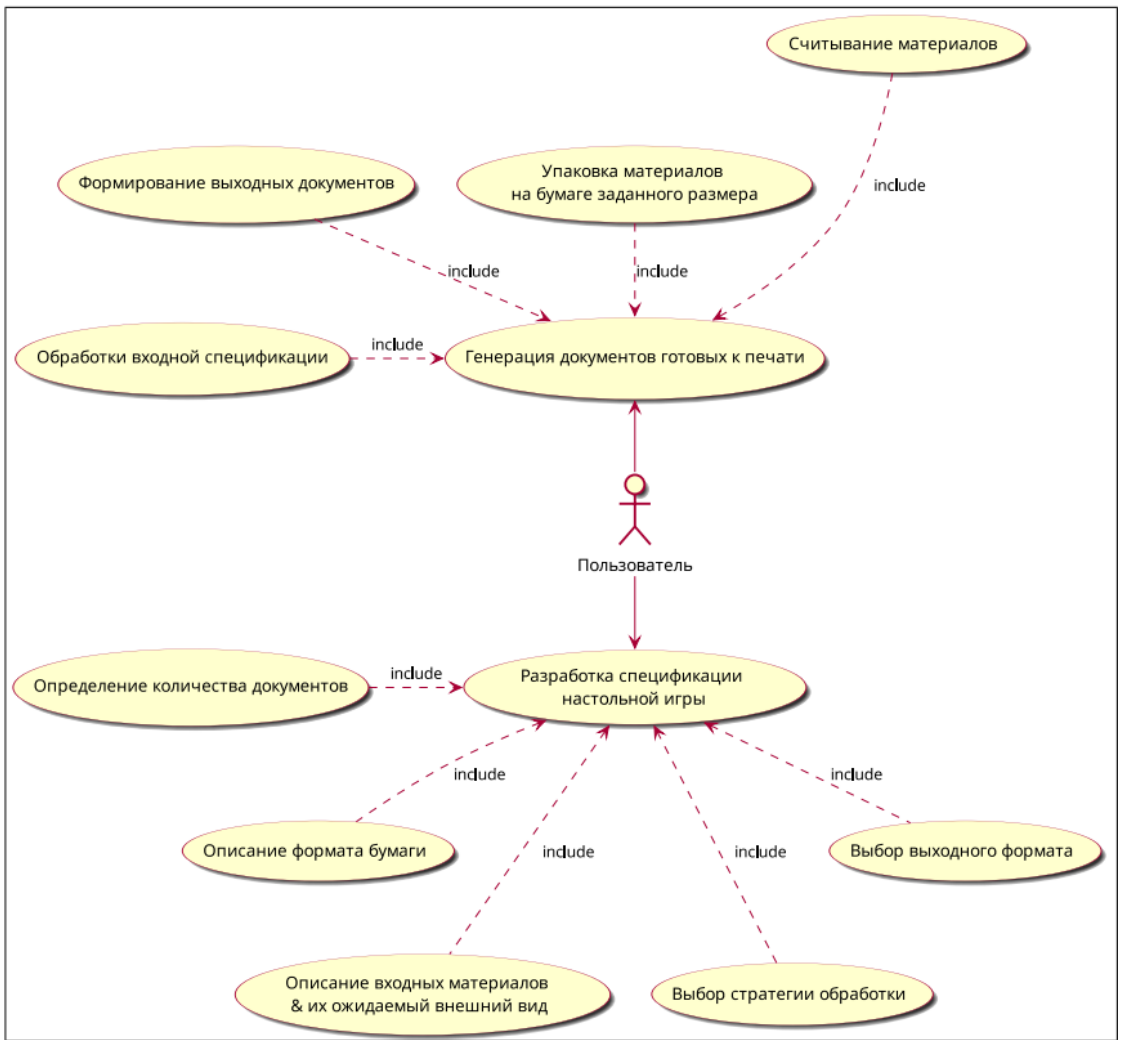


Рисунок 1 – Диаграмма вариантов использования программного средства

Из диаграммы вариантов использования (рисунок 1) видно, что все функции программного средства условно можно подразделить на две категории: одна позволяет составить спецификацию, описывающую как подготовить исходные данные и преобразовать их в документ готовый к печати, вторая позволяет совершить само преобразование входных материалов в итоговый документ.

Разберем подробнее процесс преобразования и генерации документа. Для дальнейшего моделирования понадобятся такие примитивы как точка, размер и отступы. Они в свою очередь объединяются в более комплексные сущности: обычную бумагу, рулонную бумагу и компоненты. Упаковка объектов в двумерном ограниченном пространстве осуществляется стратегиями упаковки. Для того чтобы структурировать модель, применяется шаблон проектирования «стратегия». Стратегия – это поведенческий шаблон проектирования, он определяет семейство алгоритмов, инкапсулирует каждый из них и делает их взаимозаменяемыми. Стратегия позволяет изменять алгоритмы независимо от клиентов, которые ими пользуются [6]. Стратегии упаковки возвращают упакованный документ. В нем описываются какие примитивы, где должны находится. Такой подход позволяет не привязываться к конкретному формату для экспорта, реализовав его отдельно.

Бумага, набор компонентов и стратегия упаковки позволяют представить спецификацию документа. После создания упакованного документа необходимо преобразовать его в файл или вывести на экран. Для этого еще раз используется шаблон проектирования «стратегия», но на этот раз с его помощью описываются классы для различных способов отрисовки.

Общий процесс обработки, начиная с разбора спецификации и заканчивая выходными документами можно представить в виде диаграммы состояний (рисунок 2). На ней видно, что после считывания и обработки спецификации, подготовку каждого отдельного документа можно выполнить независимо от выполнения других.

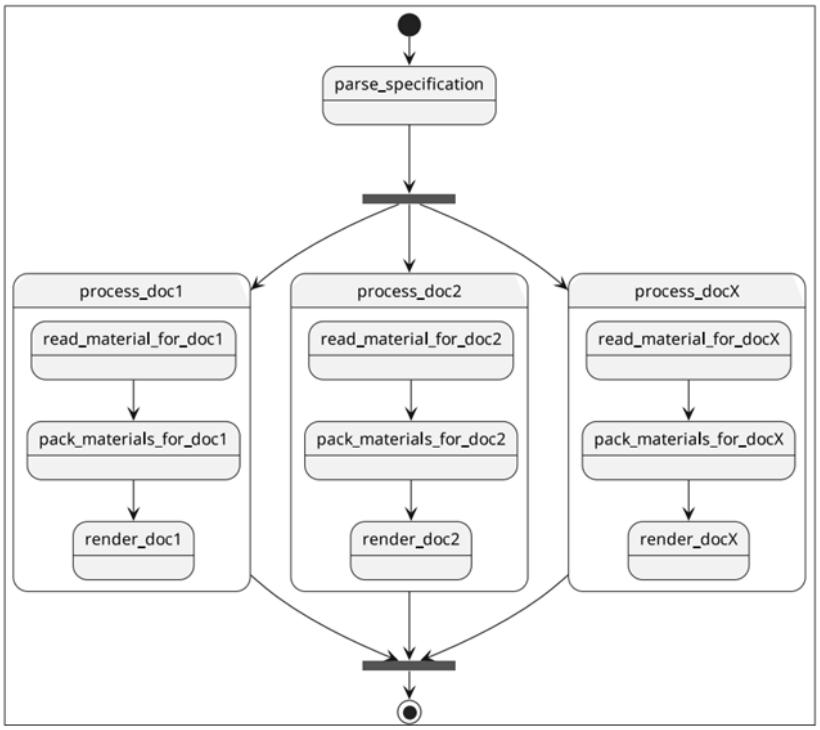


Рисунок 2 – Диаграмма состояний для генерации документа

Архитектура разработанного программного средства представляет собой три основных модуля:

— core библиотека, содержащая основные классы и функции приложения, без привязки к конкретному способу отображения;

— cli – реализация консольного интерфейса с использованием пакета click;

— gui – реализация графического интерфейса с использованием фреймворка kivy.

Консольный и графический интерфейс не зависят друг от друга, но опираются на функции и классы, предоставляемые библиотекой core. Такое разделение позволяет снизить связанность кода и уменьшает дублирование кода, позволяя реализовывать только необходимый функционал.

Библиотека может представлять собой API программного средства. Это означает что ее можно встроить в другие приложения, без необходимости устанавливать код, связанный с реализацией консольного или графического интерфейса.

Core состоит из следующих подмодулей:

— spec реализация парсера для спецификации, а также классов позволяющих манипулировать ей в объектно-ориентированном виде;

— binpack реализация различных стратегий упаковки;

— render реализация классов связанных с итоговой обработкой упакованных документов;

— pipeline реализация классов позволяющих объединить логику, представляемую подмодулями spec, binpack и render;

— measures реализация парсера и классов связанных с обработкой размеров;

— utils набор вспомогательных функций.

Обработка всех или некоторого подмножества документов, перечисленных в файле спецификации, осуществляется с помощью класса BuildPipeline. Ему на вход поступает частично обработанные данные спецификации, на основании которых осуществляется подготовка таких параметров как: формат бумаги; стратегия упаковки и рендерер.

Затем проверяет совместимость между ними, чтобы предотвратить проблемы, которые могли бы появиться на последующих этапах. Если проблем не обнаружилось, осуществляется упаковка документа на основе заданного алгоритма, а также его рендер. Обработанный документ сохраняется в виде файла или выводится на экран в зависимости от типа рендерера который использовался на предыдущем этапе.

Схема работы алгоритма конвейерной обработки документа представлена на рисунке 3а.

Перед упаковкой компонентов выполняется подготовка. Она включает в себя считывание и преобразование входных изображений фронтальной и обратной части компонентов. Схема работы алгоритма представлена на рисунке 3б.

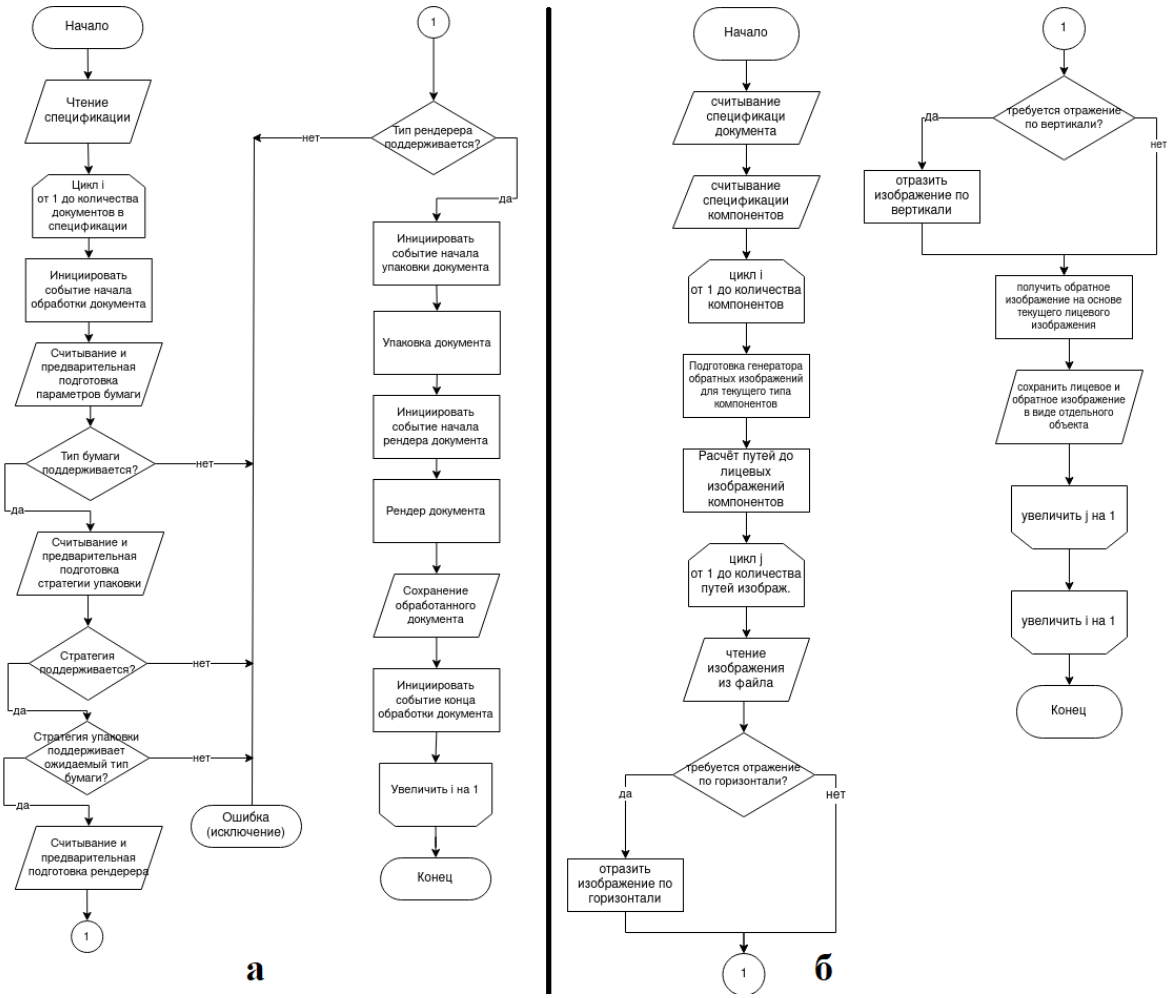


Рисунок 3 – Блок-схема алгоритмов

Во время конвейерной обработки документов необходимо находить пути к изображениям, которые должны быть вставлены в итоговый файл. Пути к ним задаются в виде двух частей, префиксной и постфиксной. Префиксы описываются в свойствах отдельного документа и представляют из себя список шаблонов путей, в которых должен начинаться поиск. Постфиксные пути же задаются в свойствах отдельного компонента. Алгоритм объединения префиксов и постфиксов представлен на рисунке 4.

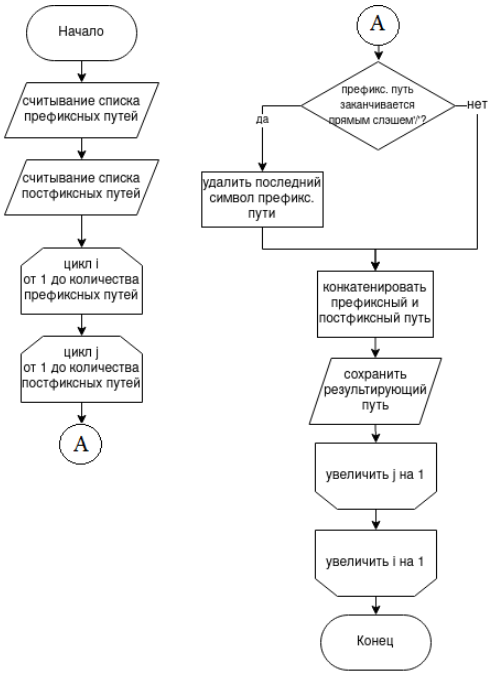


Рисунок 4 – Блок-схема алгоритма объединения префиксных и постфиксных путей

Пример работы сборки документа с предпросмотром страницы представлен на рисунке 5.

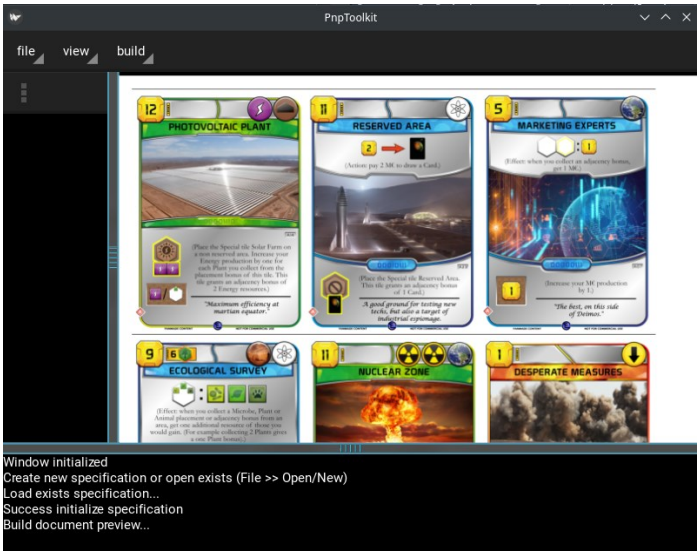


Рисунок 5 Пример сборки документа с предпросмотром страницы

***Заключение.*** Таким образом, было разработано программное средство для автоматизации верстки и подготовки к печати материалов для настольных игр, решающее задачи оптимальной упаковки в двумерном пространстве и последующей задачи гильотинного раскроя. Разработанное программное средство может быть использовано как на производстве издателями настольных игр, так и частными пользователями для подготовки и печати компонентов настольных игр способом PNP (print and play) или разработке собственных игр. Использование разработанного программного средства позволит сократить затраты на подготовку и издание или переиздание настольных игр и существенно упростить процесс подготовки и производства.

***Список использованных источников:***

*1. The Official YAML Web Site [Электронный ресурс]. — Режим доступа: https://yaml.org.*

*2. Britanica Packing Definition [Электронный ресурс].— Режим доступа: https://www.britannica.com/science/packing-combinatorics.*

*3. Nikhil Bansal, Maxim Sviridenko. New Approximability and Inapproximability Results for 2-Dimensional Bin Packing / Maxim Sviridenko Nikhil Bansal. — 2003.*

*4. Habr: про двумерную упаковку: offline алгоритмы [Электронный ресурс]. — Режим доступа: https://habr.com/en/post/136225.*

*5. Arindam Khan, Arnab Maiti. On Guillotine Separable Packings for the Two-dimensional Geometric Knapsack Problem / Arnab Maiti Arindam Khan. — 2021.*

*6. Э. Гамма, Р. Хелм. Приёмы объектно-ориентированного проектирования паттерны проектирования / Р. Хелм Э. Гамма. — Питер, 2017. — 368 с.*

UDC 004.42

# SOFTWARE TOOL FOR AUTOMATION OF LAYOUT AND PREPARATION FOR PRINTING OF MATERIALS FOR BOARD GAMES

*Zhelenok D.A, Savenko A.G.*

*Institute of Information Technologies of the Belarusian State University of Informatics and Radioelectronics, Minsk, Republic of Belarus*

*Savenko A.G. – Master of Engineering Sciences*

**Annotation.** The paper presents a software tool designed to automate the layout and preparation for printing board game materials. The software tool has a graphical and console interface, a well-thought-out architecture that allows you not to duplicate the source code of algorithms for different interfaces. The main functions of the software are: preparation of a specification file with a wide range of parameters (various types of printed materials, sheet formats (including roll printing), options for source files of board game materials and methods for their comparison, selection of a packaging strategy); generation of documents for printing in accordance with the specification. Algorithms of the software tool solve the problems of two-dimensional packing and guillotine cutting. The use of the software tool will reduce the cost of preparation and production of board games.

**Keywords.** Publication of printed products, business process, optimization tasks, two-dimensional packaging problem, guillotine cutting problem, process automation, specification file.