C4 Model in AsciiDoctor

Purpose

The purpose of this project is to show how AsciiDoc can be used to generate C4 diagrams from PlantUML, both as html and pdf.

C4 Example

The following example is based on the System Context Diagram example at the C4 Model home page. Consequently, this C4 diagram:

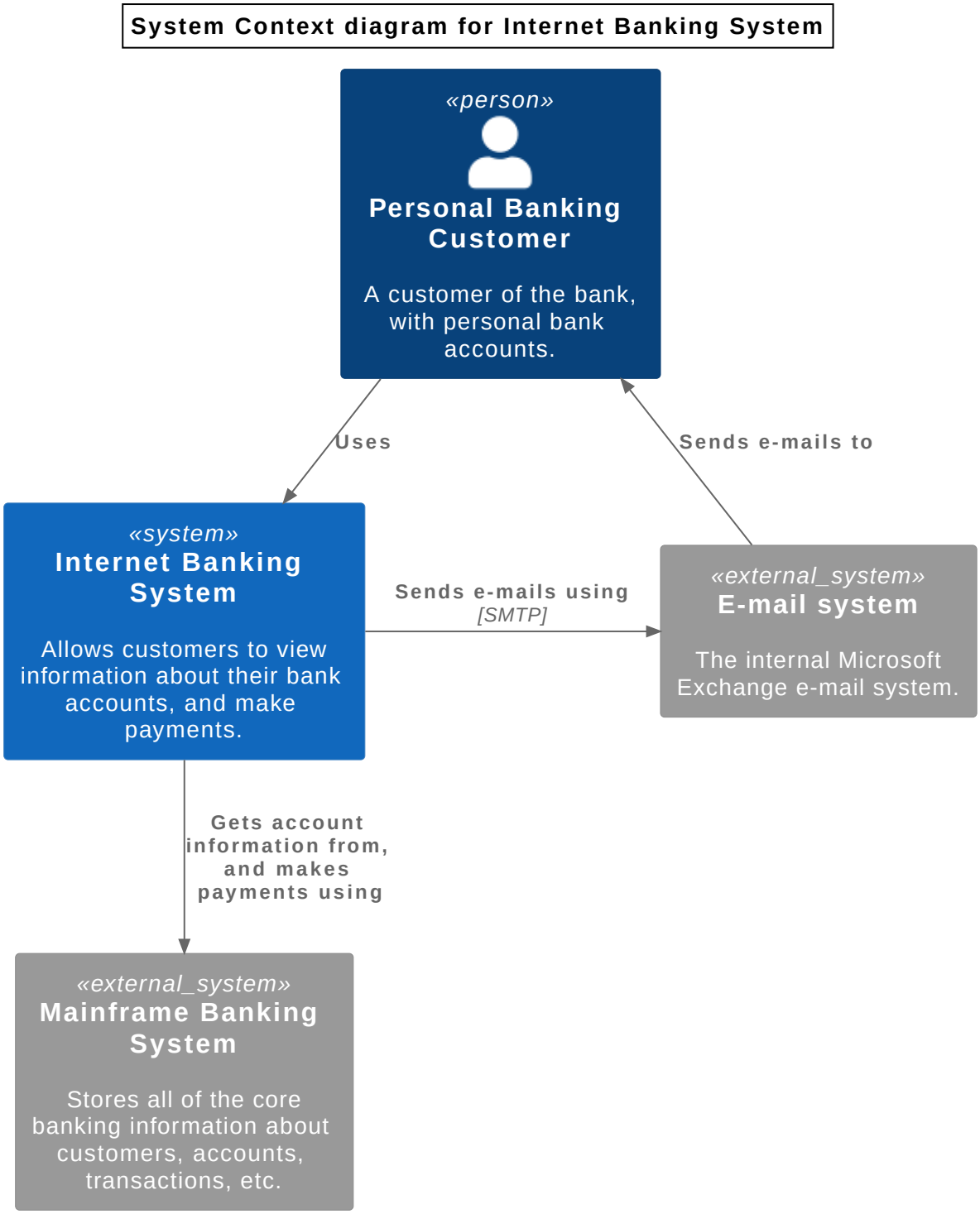

Figure 1. C4 Example

is generated from this PlantUML file:

plantuml source

```
@startuml
!include <C4/C4_Container>
title System Context diagram for Internet Banking System

Person(customer, "Personal Banking Customer", "A customer of the bank, with personal bank accounts.")
System(banking_system, "Internet Banking System", "Allows customers to view information about their bank accounts, and make payments.")

System_Ext(mail_system, "E-mail system", "The internal Microsoft Exchange e-mail system.")
System_Ext(mainframe, "Mainframe Banking System", "Stores all of the core banking information about customers, accounts, transactions, etc.")

Rel(customer, banking_system, "Uses")
Rel_Back(customer, mail_system, "Sends e-mails to")

Rel_Neighbor(banking_system, mail_system, "Sends e-mails using", "SMTP")
Rel(banking_system, mainframe, "Gets account information from, and makes payments using")
@enduml
```

Build Local

Prerequisites

- You need to have Docker installed and running.
- Download the `asciidoctor/docker-asciidoctor` image:

```
docker pull asciidoctor/docker-asciidoctor
```

Render html

This project can be rendered as html using this bash command:

generate html

```
docker run --rm \
  --volume "$(pwd)":/documents/ \
  asciidoctor/docker-asciidoctor \
  asciidoctor \
  --require asciidoctor-diagram \
  --destination-dir target \
  docs/index.adoc
```

Explanation

Line by line explanation of the command above:

1. `docker run` starts a Docker container. The `--rm` option removes the container after the command has completed.
2. `--volume` mount a volume to the container, i.e. the current directory (as returned by `$(pwd)` command) is mounted to the `/documents/` directory in the running Docker container. The `/documents/` directory is the default working director of the `asciidoctor/docker-asciidoctor` Docker image.
3. `asciidoctor/docker-asciidoctor` the name of the Docker image that is used to create the Docker container.
4. `asciidoctor` the AsciiDoctor command to render html.
5. `--require asciidoctor-diagram` AsciiDoctor argument that requires the asciidoctor-diagram to be used (necessary for PlantUML rendering).
6. `--destination-dir target` asciidoctor argument that defines the destination directory of the output files in the Docker container. In this case the `target` directory.
7. `docs/index.adoc` the name of the AsciiDoc source file to generate the file from. Only the root document needs to be included since the other files are linked from it.

Render pdf

Alternatively, this project can be rendered as a pdf:

generate pdf

```
docker run --rm \
  --volume "$(pwd)":/documents/ \
  asciidoctor/docker-asciidoctor \
  asciidoctor-pdf \
  --require asciidoctor-diagram \
  --destination-dir target \
  docs/index.adoc
```

Explanation

As can be seen, rendering a pdf is very similar to rendering html. The only difference is the asciidoctor command on line 4 that has been changed to `asciidoctor-pdf`.

Build GitHub Actions

Build using GitHub Actions and publish to GitHub Pages:

github action

```
1  name: Build and deploy
2
3  on:
4    push:
5      branches:
6        - main
7
8  jobs:
9    build:
10     runs-on: ubuntu-20.04
11
12     container:
13       image: asciidoctor/docker-asciidoctor:1.18
14       volumes:
15         # /documents/ is the default working directory of the docker-asciidoctor
   container
16         - ${{ github.workspace }}:/documents/
17
18     env:
19       BUILD_PATH: target
20       DEPLOY_PATH: _site
21
22     steps:
23       - name: Checkout □□
24         uses: actions/checkout@v3
25
26       - name: Generate html □
27         run: >
28           asciidoctor
29           --require asciidoctor-diagram
30           --destination-dir ${{ env.BUILD_PATH }}
31           docs/index.adoc
32
33       - name: Generate pdf □
34         run: >
35           asciidoctor-pdf
36           --require asciidoctor-diagram
37           --destination-dir ${{ env.BUILD_PATH }}
38           docs/index.adoc
```

```
39
40      - name: Copy artifacts 
41        run: >
42          apk add rsync &&
43          rsync -av --exclude=".*" ${{ env.BUILD_PATH }}/ ${{ env.DEPLOY_PATH }}
44
45      - name: Deploy 
46        uses: JamesIves/github-pages-deploy-action@v4.3.3
47        with:
48          branch: public
49          folder: ${{ env.DEPLOY_PATH }}
```

Explanation

Lines	Purpose
3-6	Trigger the action every time a commit is pushed to the `main` branch at GitHub.
10	Use Ubuntu as the GitHub runner environment.
12-16	Use the asciidoctor/docker-asciidoctor image so that the AsciiDoctor tools are available during the build.
18-20	Define paths used during the build as environment variables.
23-24	The checkout action used by the GitHub workflow to checkout the source code.
26-31	Use the `asciidoctor` command to generate the docs as html in the `BUILD_PATH` directory.
33-38	Use the `asciidoctor-pdf` command to generate the docs as pdf in the `BUILD_PATH` directory.
40-43	Install rsync and copy the generated files to the `DEPLOY_PATH` directory.
45-49	Use GitHub Pages Deploy Action to deploy the files in the `DEPLOY_PATH` to GitHub Pages.

References

Link	Comment
https://github.com/matsev/c4-model-adoc	This project at GitHub
https://github.com/asciidoctor/docker-asciidoctor#readme	AsciiDoctor Docker at GitHub
https://plantuml.com	PlantUML
https://github.com/plantuml-stdlib/C4-PlantUML#readme	C4 PlantUml
https://docs.github.com/en/actions	GitHub Actions

Link	Comment
https://pages.github.com	GitHub Pages