PCPT - Prolifics Code Profiler and Transformer

Introduction

This repository provides everything you need to install and use **PCPT**, a tool for analyzing, transforming, and migrating legacy applications using AI. It includes a shell script that runs the `pcpt` Docker image locally.

> ⚠ **Note:** The shell script is designed for **Linux** and **macOS** environments. It can also be made to work on **Windows** systems with a suitable Linux emulation layer (e.g., WSL2).

Contents

- `install/` – Contains all files required to set up PCPT.
- `samples/` – Sample applications to test PCPT.
- `prompts/` – Custom prompts for specific use cases. Note that these should be manually copied to `~/.pcpt/prompts/` after installation, but once you do that these will override the prompts shipped with PCPT until you remove them.
- `docs/` and `migrated/` – These folders must exist (even if empty) before running commands in the "Try it Out" section.

Versions

There are two PCPT versions:

- `stable`: Production-ready version (from the `stable` Git branch).
- `edge`: Latest features (from the `edge` Git branch).

Docker image tags align with these versions (`stable`, `edge`).

Installation

1. Run the setup commands:

Linux / macOS

```
chmod +x install/*
sudo ./install/install.sh
```

2. Edit the config file and provide your OpenAI credentials:

```
vi ~/.pcpt/config/pcpt.config
```

3. Run pcpt.sh to verify the installation:

```
pcpt.sh -h
```

Windows (via WSL)

> These steps assume you have **WSL** installed with a Linux distribution (e.g., Ubuntu) as well as **Podman**.
> WSL: https://learn.microsoft.com/en-us/windows/wsl/install
> Podman: https://podman-desktop.io/docs/installation/windows-install

1. From **PowerShell**, open WSL:

   ```
   wsl
   ```

2. Navigate to the install folder (replace path with your actual location):

   ```
   cd /mnt/c/path-to-the-installs/pcpt
   ```

3. Fix Windows CRLF line endings (no `dos2unix` needed):

   ```
   sed -i 's/\r$//' install/install.sh pcpt.sh
   ```

4. Make scripts executable:

   ```
   chmod +x install/*
   ```

5. Run the install script:

   ```
   ./install/install.sh
   ```

6. Test:

   ```
   pcpt.sh -h
   ```

Amazon Bedrock - Using Access Key

If you are planning to use PCPT with an Amazon Bedrock model you'll need to do the following:

1. Provision the model:

- An AWS account + chosen Region (e.g., us-east-2).
- Bedrock model access enabled for Anthropic Claude Sonnet 4.
- An IAM principal with Invoke permissions for Bedrock.
- An AWS access key - both its id and the key text itself

2. Install AWS CLI

https://docs.aws.amazon.com/cli/latest/userguide/getting-started-install.html

3. Configure AWS credentials using the CLI

```
aws configure
```

Enter:
• AWS Access Key ID
• AWS Secret Access Key
• Default region name (e.g., us-east-2)
• Leave output format blank

This saves credentials to ~/.aws/credentials and config to ~/.aws/config.

4. Configure ~/.pcpt/pcpt.config to use the Bedrock model

[DEFAULT]
provider=langchain
langchain_backend=bedrock
bedrock_model_id=global.anthropic.claude-sonnet-4-20250514-v1:0
aws_region=us-east-2
aws_profile=default
max_tokens=10000

Amazon Bedrock - Using SSO

1. Provision the model:

- An AWS account + chosen Region (e.g., us-east-2).
- Bedrock model access enabled for Anthropic Claude Sonnet 4.
- An IAM principal with Invoke permissions for Bedrock.

2. Install AWS CLI

https://docs.aws.amazon.com/cli/latest/userguide/getting-started-install.html

3. You will need an administrator to setup SSO for you (create a user, permissions sets, and give necessary permissions to access Bedrock).

4. Ask you administrator for the following:

- SSO start URL
- SSO region
- SSO account ID
- SSO role name

5. Create a new profile in ~/.aws/config

```
[profile bedrock-sso]
sso_start_url = <your SSO start URL - looks like https://d-
xxxxxxxxxx.awsapps.com/start>
sso_region = <your SSO region>
sso_account_id = <your SSO account ID - a 12-digit number>
sso_role_name = <your SSO role name - e.g., AdministratorAccess>
region = <your bedrock region - e.g., us-east-2>
```

6. Configure AWS CLI to use SSO

```
aws sso login --profile bedrock-sso --no-browser
```

Use the URL in a browser window for you to authenticate using the provided code. After successful authentication, the CLI will be configured to use SSO for the specified profile.

7. Configure ~/.pcpt/pcpt.config to use the Bedrock model

[DEFAULT]
provider=langchain
langchain_backend=bedrock
bedrock_model_id=global.anthropic.claude-sonnet-4-20250514-v1:0
aws_region=us-east-2
aws_profile=bedrock-sso
max_tokens=10000

8. Run the following commands in your terminal before using pcpt.sh

```
export AWS_PROFILE=bedrock-sso
export AWS_SDK_LOAD_CONFIG=1
```

Try it Out

Analyze AS400 Code (using built-in prompts)

```
pcpt.sh analyze --output docs --domain-hints as400.hints samples/as400-
sample
```

```
pcpt.sh domain-model --output docs --domain-hints as400.hints --visualize
samples/as400-sample

pcpt.sh use-cases --output docs --domain
docs/domain_model_report/domain_model_report.txt --domain-hints
as400.hints --visualize samples/as400-sample

pcpt.sh user-experience --output docs --domain-hints as400.hints --
visualize samples/as400-sample

pcpt.sh sequence --output docs --domain-hints as400.hints --visualize
samples/as400-sample

pcpt.sh business-logic --output docs --domain
docs/domain_model_report/domain_model_report.txt --domain-hints
as400.hints samples/as400-sample
```

Convert a Talend ETL job to Pyspark (using custom prompts)

Custom prompts are located in: `install/.pcpt/prompts/`

```
pcpt.sh run-custom-prompt --output docs
samples/AccountsSync_0.1/AccountsSync/items/etljobs/process/Projector_PPM
extract_job_steps.templ

pcpt.sh run-custom-prompt --input-file
docs/extract_job_steps/extract_job_steps.md --output docs
samples/AccountsSync_0.1/AccountsSync/items/etljobs/process/Projector_PPM
reimplement_as_pyspark.templ

pcpt.sh run-custom-prompt --output docs
docs/reimplement_as_pyspark/reimplement_as_pyspark.md
generate_pyspark_code.templ

cp docs/generate_pyspark_code/generate_pyspark_code.md
migrated/accounts_sync_job_v1.py
```

Known Issues

1. If you use the `--output` option and the specified directory does not exist, no output will be produced.
2. The following commands are not yet tested in the containerized version:
   - `extract-psm`
   - `generate-pim-components`
   - `generate-code`