Matplotlib

matplotlib สามารถใช้ในการตั้งค่าต่างๆ ผู้ใช้ส่วนใหญ่คุ้นเคยกับบรรทัดคำสั่งสำหรับการสร้างพล็อตและรูปภาพแบบโต้ตอบ อินเทอร์เฟซนี้มีหน้าต่างป๊อปอัปอย่างง่ายสำหรับการแสดงและจัดการข้อมูล อย่างไรก็ตาม พลังที่แท้จริงของ matplotlib คือไลบรารีการลงจุดพื้นฐาน ซึ่งเป็นระบบปฏิบัติการที่ไม่ขึ้นกับระบบปฏิบัติการและส่วนติดต่อผู้ใช้แบบกราฟิก (GUI) สามารถใช้โดยไม่มี GUI เป็นส่วนหนึ่งของเว็บเซิร์ฟเวอร์เพื่อสร้างพล็อตและรูปภาพในเอาต์พุตสำเนาที่หลากหลาย หรือฝังลงในแอปพลิเคชันขนาดใหญ่ได้โดยใช้ GUI ตัวใดตัวหนึ่ง (เช่น GTK, Tk หรือ WXwindows) ที่ทำงานบนหนึ่งในระบบปฏิบัติการหลายตัว (เช่น Windows, OS X, Solaris และ Linux)

architectural patterns/styles

Software Architecture

matplotlib แบ่งออกเป็นสามส่วนตามแนวคิด:

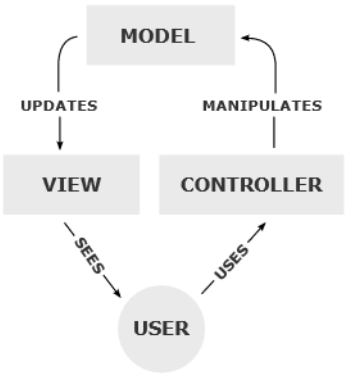
• อินเทอร์เฟซของ MATLAB คือชุดของฟังก์ชันที่อนุญาตให้ผู้ใช้สร้างพล็อตจากบรรทัดคำสั่ง

• frontend หรือ matplotlib API คือชุดของคลาสที่ทำงานหนักยกโดยการสร้างและจัดการตัวเลข ข้อความ เส้น โครงเรื่อง ฯลฯ นี่คือส่วนต่อประสานนามธรรมที่ไม่รู้อะไรเลยเกี่ยวกับผลลัพธ์

• Backend เป็นอุปกรณ์วาดภาพหรือเรนเดอร์ที่ขึ้นอยู่กับอุปกรณ์ที่เปลี่ยนการแสดงส่วนหน้าเป็นเอกสาร (JPEG, PNG, PDF, PS, SVG, Paint, GD) หรืออุปกรณ์แสดงผล (Agg, GTK/GTKAgg, TkAgg,WX/WXAgg). โค้ดการเรนเดอร์ที่สำคัญส่วนใหญ่เขียนด้วยภาษา C/C++จึงให้ประสิทธิภาพที่ดีมาก

Design Architecture

จะเป็นรูปแบบ design ของ **Model-View-Controller (MVC)**



โดยจาก software architecture นั้น จะเห็นได้ว่าเมื่อมีการเรียกใช้ไลบลารี matplot โดยตัว user นั้นจะใช้ interface ในการสั่งการจึงเปรียบ interface ได้กับ controller จากนั้นจะส่งคำสั่งไปยัง front-end ซึ่งก็คือ model จากนั้นจึง จึงจะ update ไปยัง view หรือก็คือ back-end ซึ่งจะทำหน้าที่ในการแสดงผล/render ผลลัพธ์ส่งไปยัง user

Quality Attributes Scenario

1.Usability

โดย matplotlib นั้นสามารถเรียนรู้ได้รวดเร็ว, ใช้ได้อย่างมีประสิทธิภาพ, สามารถปรับเปลี่ยนตามความต้องการของผู้ใช้ได้, สามารถทำงานได้อย่างง่ายเพราะ code ที่ใช้งานง่าย

2.Modifiability

โดย matplotlib นั้นสามารถแก้ไขดัดแปลงได้เนื่องจากเป็นsoftwareที่เป็นในรูปแบบ open-source

3. Integrability

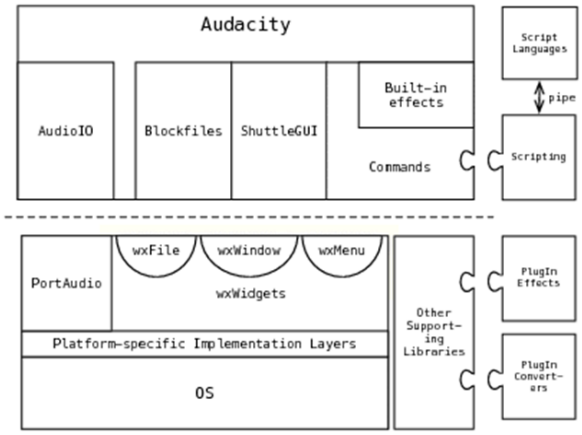
โดย matplotlib นั้นใช้งานร่วมกับไฟล์ชนิด TEX, การเซฟรูป figure เป็นไฟล์รูปชนิดมาตรฐาน

อ้างอิง: (PDF) matplotlib -- A Portable Python Plotting Package (researchgate.net)

Audacity

Audacity เป็นแอพลิเคชั่นอัดเสียงและตัดต่อเสียงซึ่งมีการใช้งานหลากหลายทั้งใน Windows และ MacOS ผู้สร้างแอพพลิเคชั่นตั้งใจที่จะสร้างแพลตฟอร์มที่ใช้พัฒนาและดีบัคอัลกอริที่มการประมวลผลเสียง หลังจาก Audacity ได้เปิดเป็น Open-Source ทำให้ผู้พัฒนาได้เข้ามาร่วมพัฒนาและแอพพลิเคชั่นก็ได้เพิ่มระบบใหม่เข้ามามากมายทั้ง ปรับปรุง บำรุงรักษา ทดสอบ อัพเดท รวมถึงทำคู่มือให้ผู้ใช้ และแปลเป็นภาษาอื่นๆ ผู้พัฒนาพยายามต่อยอดด้วยความสม่ำเสมอในด้านของลักษณะการเขียนโค้ด ด้วยการคำนึงถึงโค้ดที่อยู่ในที่ใกล้เคียง

architectural patterns/styles



Audacity มีพื้นฐานจากการใช้ Library จำนวนมาก โดย Library ที่สำคัญคือ PortAudio ที่ให้ Low-level Audio Interface และ wxWidgets ที่ให้ GUI Components

Library อื่นๆที่ใช้ต่อยอดจาก wxWidgets และ PortAudio

- BlockFile ใช้ OS file system ผ่าน wxWidgets เพื่อให้วิธีการเก็บเสียงเป็นกลุ่มเล็กๆ ซึ่งทำให้สามารถตัดต่อปรับแต่งเสียงโดยไม่จำเป็นต้องปรับแต่งทั้งไฟล์
- ShuttleGUI ใช้ wxWidgets ในการจัดการ dialog, ปุ่ม และการควบคุมอื่นๆ เพื่อการเขียนโค้ดซ้ำ ผ่านการเก็บข้อมูลเป็นตัวแปล
- Command จัดการ Bind ปุ่มในคีย์บอร์ด ผ่าน wxWidgets
- AudioIO จัดการการเคลื่อนย้ายเสียงระหว่าง Sound card, memory, hard disk ผ่าน PortAudio

Design Architecture ที่ใช้

Service-Oriented Architecture

จากโครงสร้างตามรูปด้านบนมีการใช้งาน API หลายชนิดที่มีความสัมพันธ์กัน เช่นการใช้งาน BlockFile ผ่าน wxWidget หรือการใช้ AudioIO ผ่าน PortAudio

Quality Attributes Scenario

1.Usability

จากจุดมุ่งหมายของตัว open-source ที่ต้องการให้ application ใช้งานง่าย

2.Modifiability

การเป็น open-source และการพยายามเขียนโค้ดในลักษณะเดียวกัน

3. Performance

จากการใช้ BlockFile ให้ ไม่จำเป็นต้องแก้ไขไฟล์ทั้งไฟล์

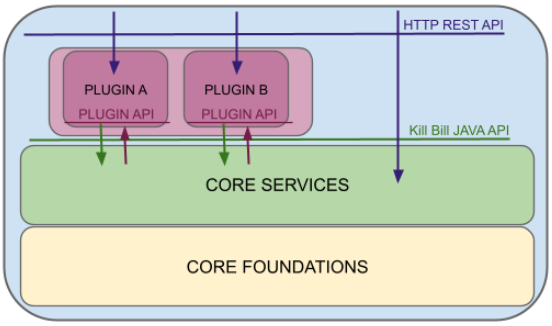
อ้างอิง : The Architecture of Open Source Applications: Audacity (aosabook.org)

Kill Bill

Kill Bill เป็น open-source แพลตฟอร์มเรียกเก็บเงินและการชำระเงินแบบสมาชิก โดยสามารถที่จะสร้าง logic ของธุรกิจตัวเองเพิ่มได้ขึ้นได้ และสามารถที่จะ custom การชำระเงินและเรียกเก็บเงินเพิ่มเติมเองได้

architectural patterns/styles

core ของ Kill Bill :



Core Services (การจัดการบัญชี, การออกใบแจ้งหนี้, การให้สิทธิ์, การติดตามหนี้, และอื่น ๆ) ถูกสร้างและบรรจุเป็น independent jars ซึ่งแต่ละบริการจะมี API เป็นของตัวเอง

Quality Attributes Scenario

1. Usability

Kill Bill มีฟังก์ชันและความยืดหยุ่นที่ให้คุณสามารถที่จะทดสอบได้ และมีความง่ายในการที่จะกำหนดฟังก์ชันของราคาและการเรียกเก็บเงินจำนวนมากที่ง่าย

- ไม่มีสัญญาณการล็อคอิน
- ไม่มีค่าสมัคร
- ไม่มีข้อจำกัดในการกำหนดคุณสมบัติ
- ไม่มีข้อจำกกัดของผู้ให้บริการบุคคลที่สาม
- ไม่มีข้อจำกัดในการสืบค้นข้อมูล

2. Modifiability

คุณสามารถเขียนปลั๊กอินเพื่อปรับเปลี่ยนการทำงานของระบบหรือผสานรวมกับผู้ขายที่เป็นบุคคลที่สามได้ เช่น การแก้ปัญหาการตรวจจับการฉ้อโกง (Accertify, Feedzai, etc.) หรือ ผู้ให้บริการด้านภาษี (Avalara AvaTax, Vertex, etc.)

3. Testability

มีการทดสอบจำนวนมาก (1,100) :

- Unit tests
- System tests
- Performance tests
- Integration tests

และมีการติดตั้งง่าย (การ run และ/หรือ การเพิ่ม tests) ในอุปกรณ์ (เช่น labtop) หรือในระบบคลาวด์ส่วนตัวหรือสาธารณะของคุณ

อ้างอิง : https://docs.killbill.io