**SWE621 – HW1**

**Group:**

Venkata Krishna Chaitanya Chirravuri **(G01336659)**

Sai Prashanth Reddy Kethiri **(G01322333)**

**Reference System –** VisTrails

**Additional Systems –** Visualization Tool Kit (VTK), matplotlib

**VisTrails**

1. **Language**
   1. **VisTrails**

Initial versions of VisTrails was written in Java and C++. Later due to increase in number of Python libraries and tools, they have changed the basis to Python.  **(Reason)** Since VisTrails aims to facilitate the orchestration of many different software libraries in workflows, a pure Python implementation makes this much easier. Many libraries written in different languages such as Fortran, C, and C++ use Python bindings to provide scripting capabilities. **(Goals)** Choosing this change of language architectural decision made them achieve goals like performance, scalability. **(Importance)** Python comprising a various amount of visualization libraries, VisTrails being a visualization tool, plotting various graphs to analyze the data would be much more efficient and simpler.

* 1. **VTK**

**(Decision) -** No, VTK did not follow the decision followed by VisTrails. InitiallyVisTrails used C++ as their base language, but later VisTrails changed their code base to Python, but VTK sticked to C++.

**(Evidence) -** <https://en.wikipedia.org/wiki/VTK>

**(Reason for difference) –** At its core, VTK is implemented as a C++ toolkit, requiring users to build applications by combining various objects into application. The system also supports automated wrapping of C++ core into Python, Java, and TCL, so VTK applications may also be written using these interpreted programming languages.

**(Difference in architectural drivers) –** Using C++ results in writing long code and inefficient use of libraries. By using Python, we can write an effective and efficient code, which can increase the performance of the system and makes the system scalable by usage of libraries.

**(Better Decision) –** I feel like VisTrails made a better decision by changing their code base from C++ to Python. Because Python gives more flexibility by making use of libraries and writing code in Python is way simpler when compared to writing code in C++. Efficiency played a major role for making this decision.

* 1. **matplotlib**

**(Decision) –** Yes, both VisTrails and matplotlib make use of Python Language.

**(Evidence) –**

<https://en.wikipedia.org/wiki/Matplotlib>

https://matplotlib.org/stable/index.html

**(Difference in decision use) –** VisTrails is a visualization tool whereas matplotlib is plotting library for Python programming language. It provides an object-oriented API for embedding plots into applications.

**(Reasons for same decision) –** Both of them chose Python as their coding platform because of efficiency in visualization.

1. **Provenance Infrastructre**
   1. **VisTrails**

Provenance refers to source or origin of objects or data. To be precise, it refers to complete information of all elements and its relationships that are responsible for existence of the data. VisTrails keeps a detail record of the steps followed and data that was extracted while following the steps. VisTrails was designed to manage multiple workflows like, it maintains provenance of data products, of the workflows that derive these products, and their executions. **(Reason)** VisTrails stores temporary results so that the user can view the results, examine the actions that are responsible for the result, can move forward or backward from the point. Provenance provides key documentation for preserving the data, ownership, determining the quality of data, and to reproducing as well as evaluating the results. **(Goals)** This architectural decision helped in satisfying the maintainability, reliability. **(Importance)** An important aspect of provenance is information about causality, i.e., a detail description of a process, along with input data and parameters, that are responsible for the creation of a data product. Thus, the structure of provenance mirrors the structure of the workflow used to derive a given result set.

* 1. **VTK**

**(Decision) –** No

**(Evidence) -** <https://gitlab.kitware.com/vtk/vtk/-/pipelines>

**(Difference in decision) –** VTK chose to follow CI/CD infrastructure. Using continuous integration and continuous delivery methodology enables to regularly deliver applications to customers by introducing automation from integration and testing phases to delivery and deployment. Implementing CI/CD pipelines helps in generating log information, so that when something goes wrong, they act as a rich source of information to understand what is happening beneath.

**(Difference in architectural drivers) –** By implementing this architecture they were able to achieve efficiency and scalability as it is easier to rollback.

**(Better Decision) –** I feel like VisTrails has a better infrastructure when compared to VTK, because the VisTrails provides information about data origin and the ownership of the data which is not available in VTK.

* 1. **matplotlib**

**(Decision) –** No

**(Evidence) –**

<https://matplotlib.org/stable/devel/testing.html>

https://docs.pytest.org/en/7.1.x/

**(Difference in decision) -** matplotlib makes use of PyTest framework. This framework allows users to write test code in Python programming language. It helps write tests for simple unit tests to complex functional tests.

**(Difference in architectural drivers) –** By using this framework it makes it easier to write test codes in an efficient and scalable manner. SO, they achieved scalability using this architecture decision.

**(Better decision) -**

1. **Usability**
   1. **VisTrails**

Besides enabling reproducible results, VisTrails holds provenance information through a series of operations and intuitive user interfaces that make the analysis of data collaboratively. VisTrails also addresses important usability issues that have hindered a wider adoption of workflow and visualization systems. **(Reason)** VisTrails keeps in mind a wide variety of users including the ones that do not have programming knowledge and provides a series of operations and user interfaces that makes workflow create, design, use and refinement. Users create and edit workflows using the Workflow Editor. Workflow graphs can be built by the user by dragging the modules from ModuleRegistry and dropping them in Workflow Editor Canvas. On selecting a module, all the parameters of the module will be displayed by VisTrails leaving the user with the options to set and modify their values. **(Importance)** Workflow interaction and result viewing can be done through VisTrails Spreadsheet. Each cell in the spreadsheet represents a view that corresponds to a workflow instance. The Version Tree View helps users to navigate through the different workflow versions. **(Goal)** The main goal achieved by this architectural decision is usability.

* 1. **VTK**

**(Decision) –** Yes

**(Evidence) –**

<https://en.wikipedia.org/wiki/VTK>

<https://github.com/Kitware/VTK/tree/master/GUISupport>

**(Implementation Difference) –** VTK has an interactive user-interface that makes it easier for a normal user to analyze and visualize the data. It makes use of vtkRenderer and vtkActor to visualize the data in 3D forms.

**(Reason for same decision) –** Makes it more user-friendly so that a user with no programming knowledge can also use the tool to visually represent the data.

* 1. **matplotlib**

**(Decision) –** In-case of user-friendly nature it is a no.

**(Evidence) -** Since it is a Python library, user must have some programming knowledge to visualize the data. Matplotlib is a graphics package for data visualization in Python.

<https://analyticsindiamag.com/comparing-python-data-visualization-tools-matplotlib-vs-seaborn/>

**(Difference in Decision) –** VisTrails and VTK on one end made use of GUI, while matplotlib on the other is a Python visualization package it requires coding from the end user to analyze and visually represent the data. This is the reason why they have not chosen GUI as their way to interact with the data.

**(Difference in architectural drivers) –** Implementing GUI makes it user-friendly and achieves easy usability. Considering the user-friendly nature, matplotlib failed to achieve usability.

**(Better Decision) –** I feel like implementing GUI is the better decision, because user can just visualize the results by click of buttons. In case of matplotlib, we need to write few lines of code to visualize the data.

1. **Versioning (Change-Based Provenance)**
   1. **VisTrails**

**(Reason)** We need to view the visual differences and compare the workflows. We also need a version tree that identifies the series of actions that are required to transform one workflow to another. **(Goals)** Implementing change-based provenance model achieved the goals of being simple and compact, making use of sustainably less space to store multiple versions of workflow. It also provides better support for reproducibility. **(Importance)** Having a version tree can support the contribution of different users. VisTrails provides a persistent storage mechanism that manages input, intermediate, and output data files, strengthening the links between provenance and data.

* 1. **VTK**

**(Decision) –** Yes

**(Evidence) –**

<https://gitlab.kitware.com/vtk/vtk/-/blob/master/Documentation/dev/git/develop.md>

**(Implementation difference) –** VTK makes use of Git, which is a powerful version control tool. VTK makes use of branchy workflow model to keep track of their versions and workflows. Here, we can update and modify our existing workflows.

**(Reason for same decision) –** With this architectural decision you could keep track of your versions and can move forward and backward in a version tree.

* 1. **matplotlib**

**(Decision) -** Yes

**(Evidence)-** <https://matplotlib.org/stable/devel/gitwash/development_workflow.html>

**(Implementation difference) –** Both matplotlib and VTK make use of GIT to maintain the workflows.

**(Reason for same decision) –** The reason is same as VTK’s reason for using GIT.

1. **Data Serialization**
   1. **VisTrails**

**(Reason)** VisTrails originally stored data in XML via simple methods embedded in its internal objects.  As the project progressed, they decided to support different serializations, including relational stores. After originally writing meta-Python, where the domain and persistence code was generated by running Python code with variables obtained from the meta-schema, they have recently migrated to Mako template. **(Goals)** By changing to Python they made use of templates that automate the code for creating classes for the schema, which provided them with the flexibility and made it scalable. **(Importance)** Automatic translation is key for users that need to migrate their data to newer versions of the system. Making use of this method for translation made it easy for developers, because a copy of each code is maintained, and translation code just needs to map one version to another. Initially, this was a forward-only map, meaning new versions could not be translated to old versions, but reverse mappings have been added for more-recent schema mappings.

* 1. **VTK**

**(Decision) –** Yes

**(Evidence) -**

<https://vtk.org/doc/nightly/html/classvtkFieldDataSerializer.html>

<https://kitware.github.io/vtk-js/api/IO_Core_Serializer.html>

https://public.kitware.com/pipermail/vtkusers/2000-May/086267.html

**(Implementation Difference) –** VTK provides the functionality of serializing and deserializing the data by making use of a concrete instance of vtkObject which is vtkFieldDataSerializer. But, compared to VisTrails it would be fairly hard to add the concept of data serialization to VTK.

**(Reason for same decision) –** The main of serializing the data in VTK is to prepare the data to transfer over Message Passing Interface (MPI) or other communication mechanisms. By implementing this architectural decision, they have achieved availability.

* 1. **matplotlib**

**(Decision) –** Yes

**(Evidence) –**

<https://docs.python.org/3/library/marshal.html>

<https://machinelearningmastery.com/a-gentle-introduction-to-serialization-for-python/>

**(Implementation Difference) –** As matplotlib is a Python’s visualization package, serialization of data is done by pickle module where it implements binary protocols for serializing and de-serializing Python object structure. Pickling is the process where Python object hierarchy is converted into byte stream, unpickling is an inverse operation.

**(Reasons for same decision) –** The reason is same as for VTK. Main advantage of using pickle is it has cross-language compatibility.

1. **Extensibility**
   1. **VisTrails**

**(Reason)** Initially VisTrails used to have fixed set of modules, but it severely limited long-term utility. VisTrails supports the VTK visualization library and VTK contains about 1000 classes, which change depending on compilation, configuration, and operating system. Since it seems, counterproductive to write different code paths for all these cases, they decided it was necessary to dynamically determine the set of VisTrails modules provided by any given package. This is when Python came into picture. **(Goals)** With the help of this decision they were able to increase the performance of system and made the system extensible and scalable. (**Importance)** Python offers a nice infrastructure for dynamically defined classes and reflection. Using Python, they were able to specify behavior of user defined VisTrails modules, removing the barrier for adoption.

* 1. **VTK**

**(Decision) –** Yes

**(Evidence) –**

https://vtk.org/Wiki/VTK/Remote\_Modules

**(Implementation Difference) –** You can achieve extensibility in VTK by creating remote modules. It is available from VTK 6. Remote modules are downloaded at CMake configuration time into the Remote module group, i.e., into the Remote directory of the repository tree. A Remote Module can be enabled by setting the target Module\_<module name> CMake configuration variable to ON just like other VTK Modules.

**(Reason for same decision) –** The reason for choosing this architectural decision is to increase the functionality of efficiency of the system. Creating remote modules can automate few processes rather than writing code multiple times for the same process to happen

* 1. **Matplotlib**

**(Decision) –** Yes

**(Evidence) –**

<https://matplotlib.org/3.1.0/faq/howto_faq.html>

<http://code.google.com/p/avoplot/>

https://openresearchsoftware.metajnl.com/articles/10.5334/jors.ai/

https://subscription.packtpub.com/book/application-development/9781847197900/1/ch01lvl1sec01/merits-of-matplotlib

**(Implementation Difference) –** matplotlib makes use of Python language which is high-level and easily extensible. matplotlib can fit every use case because it has a lot of graph types, features, and configuration options. AvoPlot is a simple-to-use graphical plotting program written in Python and making extensive use of the matplotlib plotting library.

**(Reason for same decision) –** Having multiple features and different types of graphs makes it more usable and would make it more efficient to visualize and analyze the results.

1. **Data Communication**
   1. **VisTrails**

**(Importance)** One of the features of VisTrails modules and its communication is that data is passed between VisTrails modules are themselves VisTrails modules i.e., a module can itself produce as an output of a computation. Main advantage and can also be considered as a disadvantage of the system is, these libraries allow objects to produce other objects as computational results, making a wrapping that distinguishes between computation and data more complicated. **(Reason)** Allows reasonably fine-grained control over the process. **(Goal)** The goal that this decision achieves is that defining constants and user-settable parameters in workflows becomes easier and more uniformly integrated with the rest of the system. This makes the system more accessible and improves the performance.

* 1. **VTK**

**(Decision) –** Yes

**(Evidence) -** <https://vtk.org/doc/release/5.6/html/a01189.html>

**(Implementation difference) -** vtkMultiProcessController is used to control multiple processes in a distributed computing environment. It has methods for executing single/multiple method(s) on multiple processors, triggering registered callbacks) and communication. On top of the generic communication infrastructure, vtkWeb provides protocol for vtkRenderWindow synchronization. As a result, interactions reflect both in the web page and on the server where the data lives.

**(Reason for same decision) –** Makes the processes to communicate properly and execute in a timely fashion, thereby increasing the efficiency by concurrently executing the processes. Useful for managing multiple processes at a time.

* 1. **matplotlib**

**(Decision) –** Yes

**(Evidence) -** <https://github.com/matplotlib/matplotlib/blob/cf4d8124bed4ba1bd83b3c847834577f54ca981d/examples/user_interfaces/embedding_webagg_sgskip.py>

**(Implementation Difference) –** matplotlib package makes use of WebSocket for interactive communication between the plot in the browser and the server. It requires 2 callback methods send\_json() that send JSON to the browser which is encoded as a string and sent to the socket, send\_binary() which sends binary image data to the browser.

**(Reason for same decision) –** Main purpose of making this decision is, sometimes it is not necessary to display all plots in the browser. There might be a requirement when you want to display images on the server. This is achieved by making use of matplotlib’s built-in Tornado-based server.

1. **Persistent Data**
   1. **VisTrails**

**(Reason)** Many workflow systems store filesystem paths to data as provenance, but this approach is problematic. A user might rename a file, move the workflow to another system without copying the data, or change the data contents. In any of these cases, storing the path as provenance is not sufficient. Hashing the data and storing the hash as provenance helps to determine whether the data might have changed but does not help one locate the data if it exists. To solve this problem, they created the Persistence Package, a VisTrails package that uses version control infrastructure to store data that can be referenced from provenance. **(Importance)** They make use of Universally Unique Identifiers (UUIDs) to identify the data and commit hashes from the git to reference versions. If the data changes from one execution to another, a new version is checked in to the repository. **(Goal)** This makes the system more efficient in retrieving the data and much more scalable in storing the data by making use of hashing concept.

* 1. **VTK**

**(Decision) –** Yes

**(Evidence) –**

<https://vtk.org/Wiki/VTK/InSituDataStructures>

https://vtk.org/doc/nightly/html/classvtkUnstructuredGridBase.html

**(Implementation Difference) –** Rather than reuse the vtkAbstractArray solution of creating a mapped subclass that emulates the API of the VTK object, we introduce a superclass to vtkUnstructuredGrid that removes the implementation dependant methods. The vtkUnstructuredGridBase class defines a set of pure virtual functions, taken from the vtkUnstructuredGrid API, that allow interaction with the cell information in a generic way. Combined with the cell-related methods in vtkDataSet, this provides a suitable level of abstraction for implementing a mapped dataset.

**(Reason for same decision) –** Reason for implementing this decision is that interaction and retrieval of the data can be made more efficient when compared with vtkAbstractArray. With this decision data retrieval can be more efficient.

* 1. **matplotlib**

**(Decision) –** Yes

**(Evidence) -** <https://www.tutorialspoint.com/how-to-extract-data-from-a-matplotlib-plot>

**(Implementation difference) –** In matlplotlib we can make use of in-built functions to extract the data from the results. For example, consider a graph plot. We can get datapoints using the methods get\_xdata() and get\_ydata().

**(Reason for same decision) –** Could get more insights about the data and would be helpful for analysis as well.

1. **Results**
   1. **VisTrails**

**(Reason)** According to current publications reproducing the results of a computational experiment always led to failure. The main hurdle to reproduce these results is that we again need to create a bundle that includes all components like data, code that reproduces and verifies the result. **(Importance)** VisTrails developed a package that enables results present in papers to be linked to their provenance. Using the LaTeX package users can include figures that link to VisTrails workflows. To share the results VisTrails created Crowd Labs, which is social website that integrates a set of usable tools and scalable infrastructure to provide an environment for scientists to collaboratively analyze and visualize the data. **(Goal)** Through this architectural decision they have achieved scalability and producing efficient results.

* 1. **VTK**

**(Decision) –** Yes, we can share the results of a computation in VTK.

**(Evidence)-** <https://vtk.org/Wiki/VTK/FAQ#Shared_builds_of_VTK_and_debugging_QVTKWidget_using_Visual_Studio>

**(Implementation difference) –** Results of the computation can be shared by means of Visual Studio, by setting the path to the correct build of VTK on the debugging properties of the project.

**(Reason for same decision) –** Sharing the results could help in the process of identifying the results, performing debugs, detecting errors and will also help in getting new ideas as we collaborate with others.

* 1. **matplotlib**

**(Decision) –** There is no in-built feature to share the results of a matplotlib output. Instead, you can save the output to a file and share the file with the collaborators. So, yes there is an option to share the results in matplotlib.

**(Evidence) -** <https://www.oreilly.com/library/view/python-data-science/9781491912126/ch04.html>

**(Implementation Difference) -** One nice feature of Matplotlib is the ability to save figures in a wide variety of formats. You can save a figure using the savefig() command. 

**(Reason for same decision) –** The reason is same as the above, to collaborate with other developers and perform required operations.

1. **Caching**
   1. **VisTrails**

**(Reason)** In exploratory tasks, similar workflows, which share common sub-structures, are often executed in close succession. To improve the efficiency of workflow execution, VisTrails caches intermediate results to minimize recomputation. This can be achieved by using VisTrail Cache Manager (VCM). **(Importance)** VCM schedules execution of modules in VisTrails and stores the results along with their outputs. When a VCM identifies previously computed task in a VisTrails it just looks up at cache and reduces the expensive computation time. **(Goals)** With the help of VCM they are able to optimize the data-filtering operations, thereby achieving efficiency.

* 1. **VTK**

**(Decision) –** Yes, even VTK has the concept of caching but implemented it in a different way.

**(Evidence) –**

https://vtk.org/doc/nightly/html/classvtkTemporalDataSetCache.html#details

<https://github.com/Kitware/VTK/blob/c3b0a6c4870f7890dcb690ec604e6b6a1ba5211a/IO/Exodus/vtkExodusIIReader.cxx>

**(Difference in implementation) –** VTK implemented cache by making use of classvtkTemporalDataSetCache. It consists of member functions like SetCacheSize() and GetCacheSize(), which is the maximum number of time steps that can be retained in the memory.

**(Reason for same decision) –** Instead of performing the similar task that has common substructures once again, storing the results in cache could minimize the computation time and increase the efficiency.

* 1. **matplotlib**

**(Decision) –** Yes, the output matplotlib (figures) can be cached.

**(Evidence) -** https://github.com/matplotlib/matplotlib/blob/255f468b6cd57838c6ed6bc2cc0085b4f0fd9334/tools/cache\_zenodo\_svg.py

<https://discuss.streamlit.io/t/using-st-cache-for-matplotlib-figures-or-hash-funcs-for-complex-objects/15701>

**(Difference in implementation) –** matplotlib makes use of Streamlit widget to cache the results.

**(Reasons for same decision) -** Sometimes we create a function that builds a complex Matplotlib figure out of a dataframe. So again if we want the same figure we need not rerun everything, instead we can simply interact with a Streamlit widget, so you decide to decorate it with @st.cache.