**Elemental Software Systems Software Usage**

**Module 7: Critical Thinking**

Charis Church

Colorado State University Global

MIS581: Capstone - Business Intelligence and Data Analytics

Dr. Justin Bateh

Due: June 4, 2023

**Elemental Software Systems Software Usage**

**Abstract**

Elemental Software Systems provides mechanical computational simulation software with modules in solid mechanics, structural dynamics, and thermal fluids. Understanding user behavior and usage data is critical to ensure these software products' success. The software usage dataset provided by Elemental Software Systems captures individual user runs of the modules, including job identification number, date, the module used, platform, processors, wall time, memory usage, success/failure status, version number, and capability used. Analysis of this data can help optimize software development processes, improve software quality, and enhance user experience. Additionally, understanding version numbers and capabilities used can aid in support and maintenance efforts.

Elemental Software Systems specializes in providing mechanical computational simulation software encompassing modules in solid mechanics, structural dynamics, and thermal fluids. The success of these software products hinges upon a comprehensive understanding of user behavior and utilization patterns. To this end, Elemental Software Systems has curated a rich dataset of software usage, encompassing crucial information such as job identification number, date, the specific module utilized, platform details, processor specifications, wall time, memory usage, success or failure status, version number, and specific capabilities employed. By subjecting this dataset to in-depth analysis, valuable insights can be extracted to optimize software development processes, enhance software quality, and elevate the overall user experience. Furthermore, the understanding of version numbers and capabilities employed holds significant potential in bolstering support and maintenance endeavors, ultimately contributing to the continued success of the software offerings.

**Introduction**

Elemental Software Systems at Sandia National Laboratories produces a modeling and simulation software product for United States Government research purposes.  Sandia National Laboratories is one of twelve laboratories that operate as federally funded research and development centers. Elemental Software Systems provides mechanical computational simulation software with areas involving solid mechanics, structural dynamics, and thermal fluids. These tools are used daily to simulate high-consequence events in a graphical way, such as simulations involving mechanical stresses in passenger aircraft.  Other users use these simulations within Sandia National Laboratories to safeguard the safety of products designed and manufactured by the laboratories and ensure that the laboratories always provide exceptional service in the national interest.  Such modeling simulations are most typically performed on high-performance computers. These simulation tools are key to fulfilling that mission - Mirageblade provides solid mechanics simulations, Phoenixstorm offers structural dynamics simulations, and Crimsonfire offers thermal fluid simulations. These three modules together make up the Elemental Software Systems.

The Elemental Software Systems works within an agile software development framework. Agile is a methodology that focuses on a self-organizing team that desires to continuously improve and deliver by using adaptive planning. The social practices of agile include pair-programming, daily stand-ups, and retrospectives. Additionally, the two main roles that are defined on the team are the product owner and the scrum master. The product owner is responsible for the project outcomes and has primary engagement with the stakeholders. The scrum master is a servant leader focusing on the team’s effectiveness. The technical practices of an agile team are unit testing, continuous integration, refactoring, collective ownership, and code standards (Gupta, George, & Xia, 2019). One of the types of agile that are used the most within Elemental Software Systems is Scrum. Scrum provides a framework for meetings, tools, and roles that work together to help teams manage their work. Scrum operates on small cycles, called sprints, to achieve increments of work. Elemental Software Systems operates in three-week sprints. During these cycles, teams pull from their product backlog and place it into a backlog for that sprint where the team works on those items to produce a potentially usable product increment by the close of the sprint. One of the parts of the sprint process is a sprint review which provides the information once a sprint to the customer on the increment of work that was produced, and what is the next planned increment.

**Problem Framing**

Even though Elemental Software Systems has been developing codes for almost ten years, and the group has been agile for the last five years, the group has only started to increase its use of business intelligence within the last few years. Not understanding the users and how they use the codes leads to not being able to make the right resource allocations. Knowing the number of users can help managers allocate resources effectively. For example, if the user base is growing rapidly, managers may need to invest in additional servers, bandwidth, or customer support personnel to meet increased demand. Conversely, if the user base is shrinking, managers may need to reassess their resources and find ways to optimize costs.

Second, there is no strategic product development occurring. Currently, significant resources are being invested into developing new features that are suspected to be not aligned with user needs, and there are concerns that there is low adoption and usage as a result. The number of users can guide product development efforts. By understanding the size and characteristics of the user base, managers can identify areas for product improvement or new feature development. If the user base is predominantly using a specific feature of the product, managers can prioritize enhancing that feature to better meet user needs and preferences.

Additionally, it is unclear if the right strategic decisions are being made to ensure there is proper user support. There are currently three individuals doing user support but the load for these staff is overwhelming. It is unclear if the workload needs to be prioritized or if more staff need to be hired to do user support. Managers would be able to use usage data information to anticipate and address potential issues related to customer inquiries, complaints, or feedback. They can also identify patterns or trends in user behavior, satisfaction levels, and preferences, which can help improve customer support processes and overall customer experience.

Next, there is currently no financial planning and budget forecasting related to the number of code users. The budgets have only previously been based on historical decisions. The number of users can be used in financial planning and forecasting. Managers can use this information to estimate revenues, cash flows, and profitability, which can inform budgeting, financial projections, and business valuation. It can also help in evaluating investment opportunities, assessing pricing strategies, and making informed financial decisions.

Furthermore, there is no understanding of where the company is about its competition. Knowing the number of users can provide insights into the organization's competitive position. Managers can compare their user base with that of competitors to assess relative market share, identify areas for growth, and stay ahead of the competition. This information can help managers formulate competitive strategies, benchmark performance, and track progress over time.

Finally, there is no understanding of how the codes are being used. The number of users can help managers optimize resources and operations. By analyzing user data, managers can identify usage patterns, peak usage times, and resource-intensive activities. This information can be used to optimize operations, such as server utilization, staffing levels, and process efficiencies, resulting in improved resource allocation and cost savings.

Addressing these problems through effective problem framing and leveraging the valuable dataset on code usage can empower Elemental Software Systems to make data-driven decisions, improve software quality, enhance user experience, and ultimately achieve their software development goals. Reviewing this information can help address various problems related to software development, software maintenance, and software quality assurance. By implementing and effectively leveraging the rich dataset on code usage, Elemental Software Systems can gain valuable insights into its software development processes. This can enable them to identify patterns, trends, and areas of improvement, leading to data-driven decisions that can positively impact software quality. Furthermore, by using this dataset, Elemental Software Systems can proactively address issues related to software maintenance, identify bottlenecks, and optimize their software development workflows. This holistic approach can enhance user experience, reduce defects, and improve overall software quality assurance. Ultimately, leveraging the power of data-driven decision-making can empower Elemental Software Systems to achieve its software development goals with greater efficiency and effectiveness.

**Objectives**

It is critical to understand users and how they use products to ensure the software product’s success (Deng, Wellsandt, Hribernik, & Thoben, 2021, p. 3299). As organizations increasingly rely on software for various purposes, software usage datasets data play a crucial role in optimizing software development processes and achieving organizational goals. Elemental Software Systems aims to leverage the rich dataset of software used to make data-driven decisions, improve software quality, enhance user experience, and optimize software development workflows. By analyzing the data and gaining insights from patterns and trends, they can inform decision-making processes, identify areas of improvement, and align software development strategies with organizational goals. In this rapidly evolving landscape of software development, leveraging the power of data in the form of software usage data can empower managers to drive effective software development practices and achieve successful outcomes.

A few goals are data-driven decision-making and software quality improvements. The Elemental Software Systems dataset of software usage uses data to inform decision-making processes related to software development strategies, resource allocation, feature prioritization, and overall product roadmap. By analyzing the data, they can make informed decisions based on patterns and trends observed in the dataset, leading to more effective and efficient decision-making. Additionally, software usage is key to identifying areas of improvement in software quality. This includes analyzing the data to detect and address recurring defects, reduce technical debt, and optimize software performance. By leveraging the dataset, managers can gain insights into software quality issues and make data-driven recommendations to improve overall software quality.

Some other goals would include improving the user experience and optimizing the workflow. Analyzing the dataset to gain insights into how users are interacting with the software drastically improves the user experience. This includes identifying pain points, usability issues, and customer satisfaction metrics. By leveraging the dataset, managers can make data-driven recommendations to enhance user experience, usability, and overall customer satisfaction. Each user may have different workflows for the same software and may vary from person to person and the purpose of the engagement with the software (Andrei & Calder, 2018, p. 195). Optimizing software development workflows will align software development strategies with organizational goals. The data can identify bottlenecks, inefficiencies, and areas that require maintenance or updates. This helps in streamlining workflows, optimizing development processes, and making informed decisions about future software development directions and investments.

In summary, the goals of the software usage dataset include data-driven decision-making, software quality improvement, user experience enhancement, and workflow optimization, all of which contribute to the overall success of software development and aligning software development strategies with organizational goals. The effective utilization of the dataset of software usage empowers managers to make informed decisions, optimize software development practices, and align software development strategies with the overarching goals of the organization. By leveraging the insights gained from the dataset, managers can drive continuous improvement in software quality, enhance user experience, and streamline development workflows, leading to more efficient and effective software development processes. The strategic use of the dataset of software usage enables managers to leverage data-driven decision-making to enhance software development outcomes, ultimately resulting in improved software products, increased customer satisfaction, and organizational success in the competitive software development landscape.

**Overview**

Elemental Software Systems provides mechanical computational simulation software encompassing solid mechanics, structural dynamics, and thermal fluids. Their software modules, Mirageblade, Phoenixstorm, and Crimsonfire, collectively form Elemental Software Systems. Understanding user behavior and usage patterns is crucial for the success of software products. To achieve this, Elemental Software Systems seeks to leverage a vibrant dataset of software usage, enabling data-driven decisions, software quality improvement, enhanced user experiences, and optimized software development workflows. The dataset captures valuable information such as job identification numbers, execution dates, module usage, platforms, processors, wall time, memory usage, success/failure status, version information, and capabilities used.

The first research question focuses on understanding the typical usage patterns for the codes present in the dataset. By analyzing the dataset, this research question can provide insights into code frequency, platform performance, software development bottlenecks, and security vulnerabilities. The second research question delves into determining the most frequently utilized capabilities within the organization and their impact on overall performance. By inspecting the dataset, this research question aims to identify critical capabilities, allocate resources effectively, replicate best practices, optimize costs, and gain a competitive advantage. Examining differences in capabilities and performance impact can be helpful to determine strategy. Elemental Software Systems' dataset provides an opportunity to explore usage patterns, platform performance, and capability impact within the organization. By answering the research questions and testing the associated hypotheses, valuable insights can be gained regarding software development processes, resource allocation, compatibility, performance optimization, quality assurance, and security.

**Research Questions and Hypotheses**

It is critical to understand users and how they use products to ensure the software product’s success (Deng, Wellsandt, Hribernik, & Thoben, 2021, p. 3299). As organizations increasingly rely on software for various purposes, software usage datasets data play a crucial role in optimizing software development processes and achieving organizational goals. Elemental Software Systems aims to leverage the rich dataset of software used to make data-driven decisions, improve software quality, enhance user experience, and optimize software development workflows.

This dataset provides information related to individual user runs of the various modules, including a specific job identification number, the date it was run, the module used, and the platform. It also includes details on the number of processors, wall time, and memory usage. Processors can refer to either hardware components or software programs that execute instructions and perform computations. The speed and efficiency of processors have a significant impact on software performance and usability. Therefore, choosing the right hardware or optimizing software programs for efficient processing is crucial for delivering a satisfactory user experience.

Wall time in software refers to the actual time it takes for a program to execute from start to finish, including waiting time for input/output operations to complete. Measuring wall time helps identify bottlenecks and inefficiencies in program execution, enabling developers to optimize performance and enhance the user experience. Memory usage, on the other hand, describes the amount of computer memory (RAM) used by a software program. Monitoring memory usage is important because excessive consumption can negatively impact system performance and lead to software crashes.

Apart from performance related information, the dataset also provides details on the success or failure of each run. Analyzing this data helps gain a better understanding of user needs and preferences, which ultimately leads to improved software and enhanced user experiences. Additionally, the dataset includes information on the version and version number of the software used. Understanding version numbers and release information is crucial in software development for tracking changes over time, identifying introduced features and bug fixes, and quickly addressing issues. Version information also aids in support and maintenance by identifying the specific software version running on a user's system.

Finally, the dataset captures the capability that was used. This information is valuable for understanding which parts of the code are actively utilized by users and which areas are not being used. Analyzing usage patterns helps identify areas of the code that may require enhancement or optimization and highlights sections that can potentially be deprecated due to lack of use. This understanding contributes to the ongoing development and maintenance of the software, ensuring that resources are allocated efficiently, and user needs are met effectively.

**Research Questions**

Research questions are essential in guiding a research study by providing a clear focus and direction (O'Leary, 2021, p. 41). They help researchers to define the scope of the study, identify relevant variables, and develop a framework for data analysis. Moreover, research questions provide a structure for organizing and interpreting data, making it easier to draw meaningful conclusions and make informed decisions. They also help researchers to avoid bias and ensure that the study is objective and unbiased. Research questions are critical in ensuring that a research study is well-designed, well-executed, and provides valuable insights that can inform decision-making. There are two main research questions to be reviewed with this dataset. They are:

* What are the typical usage patterns for the codes present in the dataset?
* Which capabilities are being utilized most frequently within the organization, and what is their impact on overall performance?

The first question explores how frequently the codes are used, the range of platforms and processors they are executed on, and their success or failure rates. This could help the organization identify any potential issues or bottlenecks in their software development processes related to these codes and optimize their usage accordingly. Understanding which codes are used most frequently and for what purpose can help the organization identify areas of high priority or critical importance in their software development processes. This can inform decisions about resource allocation, such as which code to prioritize for further development or optimization. Next, examining the usage patterns of different codes across platforms and processors can provide insights into how the codes perform in different environments. This can help the organization identify potential issues related to compatibility or performance and take steps to address them. Additionally, analyzing the success and failure rates of different codes can provide insights into areas where the organization may need to focus on improving its software development processes. For example, if one code has a significantly higher failure rate than others, this could indicate a need for additional testing or debugging. Finally, understanding the typical usage patterns of different codes can also help the organization identify any potential security vulnerabilities. For instance, if a code is frequently used for a specific purpose that involves sensitive data, this may require additional security measures. As a result, analyzing the typical usage patterns for the codes present in the dataset can provide valuable insights into various aspects of the organization's software development processes, including resource allocation, compatibility and performance, quality assurance, and security.

The second research question is important to help understand which capabilities are most critical to the organization's success and whether those capabilities are being utilized effectively. By analyzing the capability use column in the dataset, managers can gain insights into which capabilities are being used most frequently and how they are impacting overall performance. This analysis can help managers to make better decisions about resource allocation, such as by investing in additional resources to support critical capabilities or prioritizing the development of new capabilities that are in high demand. First, knowing which capabilities are being used most frequently can help managers allocate resources, including personnel and technology, more effectively. If certain capabilities are being used significantly more than others, it may indicate a need for additional resources or training to optimize their use. Next, understanding the impact of different capabilities on overall performance can help organizations identify areas where they can improve. By analyzing the usage patterns of capabilities that have a positive impact on performance, organizations can identify best practices and replicate them in other areas of the organization (Unahalekhaka & Bers, 2021, p. 3). Also, some capabilities may be more resource-intensive than others, and understanding their impact on performance can help organizations optimize costs. By identifying capabilities that are resource-intensive but have little impact on performance, organizations can make informed decisions about whether to continue investing in those capabilities. Finally, by identifying capabilities that are critical to the organization's success, managers can leverage them to gain a competitive advantage. For example, if certain capabilities are unique to the organization and provide a significant performance advantage, they can be used to differentiate the organization from its competitors. By gaining a deeper understanding of the most used capabilities and their impact on overall performance, organizations can make informed decisions about resource allocation, strategic planning, and process optimization, leading to improved efficiency and productivity. A hypothesis is a tentative explanation or prediction for a phenomenon that can be tested through empirical research. Meanwhile, a null hypothesis is a statement that suggests there is no significant relationship between two variables or no significant difference between groups, and it serves as a benchmark for statistical testing (McNulty, 2022, p. 2).

**Hypotheses**

Hypotheses are important because they provide a clear and testable statement of what a researcher believes will happen in a study or experiment. By formulating a hypothesis, researchers can specify the variables to be tested and the expected relationship between them. This helps to focus the study and guide the data analysis process. Additionally, a hypothesis can be used to guide the design of an experiment or study, helping to determine the appropriate sample size, measurement methods, and statistical tests to use. Ultimately, hypotheses are critical for scientific inquiry because they provide a way to test and validate theories about the natural world.

**Research Question: “What are the typical usage patterns for the codes present in the dataset?”**

The first research question “What are the typical usage patterns for the codes present in the dataset?” provides the hypothesis:

* H1: The code "Mirageblade" is used more frequently than the other two codes and is more likely to be successful when executed on Qualcomm Snapdragon processors than on other platforms.
* H2: There are no significant differences in the frequency and success rate of code usage across different platforms, and the three codes are used equally often in the organization.

This hypothesis is important because it provides a clear and testable statement about the expected relationship between the code Mirageblade and the platforms it is executed on. By stating that this code is expected to be used more frequently and more successfully on Qualcomm Snapdragon processors, the hypothesis sets up a specific expectation that can be tested with the available data. The null hypothesis provides a contrasting statement that assumes there are no significant differences in the frequency and success rate of code usage across different platforms, and that the three codes are used equally often in the organization. This allows researchers to compare the observed data against the expected results under the null hypothesis and determine whether there is enough evidence to reject it in favor of the alternative hypothesis.

To test this hypothesis, a common statistical test used is the Analysis of Variance (ANOVA) since there are multiple codes. ANOVA can help answer these hypotheses by examining the differences in the mean usage frequency and success rates of the three codes across different platforms. ANOVA can test for significant differences between multiple groups (in this case, three codes) based on a continuous outcome variable (in this case, frequency and success rate) across multiple levels of a categorical variable (in this case, different platforms).

For H1, ANOVA can help determine if there are significant differences in the frequency and success rate of code usage across different platforms and if "Mirageblade" is more likely to be successful on Qualcomm Snapdragon processors compared to other platforms. This can help identify which platform is the best for executing "Mirageblade" and optimize the usage of the code accordingly. For H2, ANOVA can test if there are significant differences in the frequency and success rate of code usage across the three codes. If there are no significant differences in usage frequency and success rates between the three codes, then H2 would be supported, indicating that the codes are used equally often in the organization. SAS Studio provides the ability to run a one-way ANOVA test.

**Research Question: “Which capabilities are being utilized most frequently within the organization, and what is their impact on overall performance?”**

For the second research question, “Which capabilities are being utilized most frequently within the organization, and what is their impact on overall performance?” the hypothesis is:

* H1: The capability of "Nonlinear analysis" is being utilized most frequently within the organization and is positively impacting overall performance.
* H2: There are no significant differences in the frequency of utilization and impact on overall performance among different capabilities within the organization.

By analyzing this, the information can provide insights into the effectiveness of specific capabilities in achieving desired outcomes. By examining the relationship between capability usage and performance, the organization can identify which capabilities are most important for achieving its goals and prioritize their development and utilization. This can lead to more efficient resource allocation and better overall performance. Additionally, if the hypothesis is proven false, it could indicate that the organization should focus on improving other factors that contribute to task performance.

To test this hypothesis, an ANOVA test should be used since it will be comparing 14 different variables that are provided as options for the “capability use” attribute. ANOVA can help answer the hypothesis by testing significant differences in the mean frequency of utilization and impact on overall performance among the 14 different capabilities. This would involve comparing the mean frequency and impact scores for each capability, and determining if there are statistically significant differences between them. If H1 is supported, the results would indicate that Nonlinear analysis is being utilized more frequently and is positively impacting overall performance. If H2 is supported, the results would indicate that there are no significant differences in the frequency of utilization and impact on overall performance among the 14 different capabilities. SAS Studio provides the ability to run a one-way ANOVA test.

**Literature Review**

Elemental Software Systems intends to leverage the rich dataset of software used to make data-driven decisions, improve software quality, enhance user experience, and optimize software development workflows. Given this, there are several trends in the literature regarding agile software development and data analytics that are relevant to help better understand this usage data. Among the literature, it was discovered that there is a lack of standardization in agile software development with varying levels of growth and sophistication. While analytics is already being used in different areas of agile software development, it is not yet fully integrated or standardized across the domain.

One trend is the increasing utilization of product usage information throughout the product life cycle. The product life cycle data can provide valuable insights into user-product interactions and can inform various product development tasks (Meyer, Wiederkehr, Koldewey, & Dumitrescu, 2021). However, challenges related to information quality, data acquisition, personal data protection, and knowledge discovery need to be addressed to fully leverage this data (Deng, Wellsandt, Hribernik, & Thoben, 2021). Additionally, proactive communication with users is essential to address their concerns and ensure their trust in sharing product usage information. By transparently explaining the benefits of data collection and the measures taken to protect user privacy, companies can foster a mutually beneficial relationship with their customers, encouraging active participation in data-sharing initiatives.

Additionally, the analysis highlights the need for incorporating customer reviews as a complementary data source to usage data. Customer reviews can provide additional perspectives and insights into user experiences and satisfaction, enhancing the understanding of product performance and identifying usage-centric improvements (Biesialska, Franch, & Muntés-Mulero, 2021). Furthermore, integrating sentiment analysis techniques into the analysis of customer reviews can enable companies to extract valuable qualitative information at scale. By automatically categorizing and quantifying sentiments expressed in reviews, companies can identify recurring themes, sentiment trends, and potential areas of improvement, allowing for data-driven decision-making in product development.

Exploring the literature also emphasizes the importance of data-driven product planning, where usage data captured from sensors, user behavior, and environmental factors are used to objectively quantify product performance and drive improvements (Meyer, Wiederkehr, Koldewey, & Dumitrescu, 2021). However, challenges exist in creating a data strategy aligned with corporate objectives, integrating data analytics into traditional design processes, and making data analytics accessible to small and medium-sized enterprises. Additionally, enabling access to data analytics for small and medium-sized enterprises is an important consideration. Small and medium-sized enterprises often face resource constraints and may lack the expertise or infrastructure to implement advanced analytics solutions (Meyer, Wiederkehr, Koldewey, & Dumitrescu, 2021). Therefore, developing user-friendly tools, providing training and support, and promoting collaborations with analytics service providers can help bridge this gap, allowing small and medium-sized enterprises to leverage product usage data and stay competitive in the market.

Finally, the research suggests the application of a Scrum Reference Ontology (SRO) to support the integration of applications for data-driven decision-making in agile software development (Santos Júnior, Barcellos, Falbo, & Almeida, 2021). Currently, data-driven decisions are not widely practiced, representing missed opportunities for informed decision-making. The SRO can serve as a common conceptual framework, addressing semantic conflicts and enabling the development of integrated data-driven solutions within the Scrum context. Moreover, promoting a cultural shift towards data-driven decision-making within agile software development teams is essential to fully harness the benefits of the Scrum Reference Ontology (SRO). This involves fostering a data-driven mindset, encouraging collaboration between data scientists and software developers, and incorporating data analysis into the agile development process. By embracing data-driven practices, teams can leverage the SRO to enhance decision-making and drive continuous improvement. Overall, the findings highlight the need for standardization, improved methodologies, and broader adoption of data analytics in agile software development.

**Research Design**

Research design plays a pivotal role in any study as it serves as the blueprint for the entire research process. It provides the structure and framework for gathering and analyzing data, ensuring that the research objectives are effectively addressed. A well-designed research study helps to ensure the validity and reliability of the findings by minimizing biases and errors. It involves making important decisions about the selection of appropriate research methods, data collection techniques, and data analysis procedures. A robust research design also facilitates the replication and verification of results by other researchers, contributing to the overall body of knowledge in the field. Moreover, a carefully crafted research design allows researchers to control and manipulate variables, establish cause-and-effect relationships, and draw meaningful conclusions from the collected data. In conclusion, research design is of utmost importance as it lays the foundation for conducting rigorous and credible research, enabling researchers to obtain accurate and meaningful insights into the phenomena under investigation.

**Methodology**

The primary analysis was done using SAS to understand the research questions of “What are the typical usage patterns for the codes present in the dataset?” and “Which capabilities are being utilized most frequently within the organization, and what is their impact on overall performance?” will be answered. This will help to determine if any of the data is statistically significant. Statistical significance is crucial as it validates research findings by determining if observed differences or relationships are likely to be genuine rather than due to chance. This data is then able to help in decision-making, generalization, and avoiding false conclusions, ensuring that meaningful and reliable conclusions can be drawn from the data.

Next, SAS Studio will be used to help with predictive modeling. Predictive models will be used to forecast the future time it takes for a program to execute (wall time) based on the platform being used. Similarly, information regarding processors can be used to identify what might be used in the future. This would help coordinate with those who are buying the new platforms to make sure that these capabilities are present in future purchases. There is currently no information collected from the users regarding feedback, comments, or requests, but SAS Studio would be able to assist with that if that information was later added.

Finally, Microsoft’s Power BI will be used to visualize a dashboard of the most important views that could then be provided to the managers of the group regularly. Visualizing usage data on a dashboard is helpful because it allows developers to quickly identify patterns and insights, track key performance indicators, and make data-driven decisions. A line chart showing trends in wall time or average memory usage over time would be able to help developers identify performance issues or track improvements made to the software over time. Next, a pie chart showing the distribution of users by code module can help developers and managers better understand user needs and prioritize development efforts accordingly. A heatmap or scatterplot showing the correlation between different attributes, such as wall time and processors used, can help identify patterns or relationships that may be missed with other visualization techniques. Finally, a list showing recent user activity can help developers track user engagement, identify common issues or bugs, and respond to user feedback promptly.

**Methods**

The methodology for this analysis is structured as follows. Firstly, the research objectives are defined, which include optimizing software development processes, improving user experience, and making data-driven decisions based on the software usage dataset. The data collection process involves capturing variables such as job identification number, date, module used, platform, processors, wall time, memory usage, success/failure status, version number, and capability used. To ensure data quality, preprocessing steps are conducted, including cleaning the dataset, handling missing values, removing outliers, and standardizing variables.

The analysis process incorporates various techniques and tools, including descriptive statistics calculated using SAS, data visualization through pie charts and bar charts, one-way ANOVA for statistical analysis, and the development of predictive models using SAS Studio. Descriptive statistics provide a quantitative summary, while data visualization aids in identifying trends and hidden patterns. One-way ANOVA enables comparisons across categories to understand the impact of capabilities on overall performance. Predictive models forecast future wall time, aiding in resource planning and platform purchases. These analytical approaches offer valuable insights into usage patterns, capability impact, and software performance, enabling informed decision-making and strategic alignment in software development.

To visualize the findings, Microsoft Power BI is utilized to create a dashboard. The dashboard includes visualizations regarding what type of version was used. This provides the information to understand if users are using the most stable version of the code. A line chart provides a depiction of trends with the usage month to month. Pie charts showcase the distribution of wall time by the platform which helps to inform which platforms are being used the most. This can then ensure that the utilization of these platforms is too high and provide an understanding of future platform purchases that may have similar capabilities. The treemap illustrates correlations between different attributes, revealing patterns and relationships. By following this methodology, the analysis aims to achieve the stated research objectives and provide valuable insights for software optimization and user satisfaction.

**Limitations**

The software usage dataset and its analysis have several limitations. Firstly, the dataset has a limited scope, focusing only on individual user runs of software modules provided by Elemental Software Systems. This narrow focus may not comprehensively understand overall software usage patterns and trends. Secondly, the dataset lacks contextual information that could offer deeper insights into user behavior, motivations, and preferences. While it includes technical details like job identification numbers, dates, and hardware/software specifications, important contextual information may be missing. Furthermore, the dataset's representativeness is limited to the specific user base and usage scenarios of Elemental Software Systems. The findings and optimizations derived from this dataset may not apply to other software systems or user populations. The dataset's error analysis is also incomplete, as it does not provide detailed insights into the reasons for failures or specific error codes. This limitation hinders a thorough understanding of user experience issues and software stability.

Additionally, the dataset does not include direct user feedback or qualitative data on user satisfaction, usability, or specific feature requests. This absence of user input restricts a comprehensive understanding of user needs and preferences. Privacy concerns may arise depending on how the dataset was collected. Safeguarding user privacy and ensuring data anonymization should be a priority when working with software usage datasets. Finally, the dataset's information may be limited to a specific timeframe, which may restrict the analysis of long-term trends or changes in user behavior over time. Considering these limitations is crucial when interpreting the results and drawing conclusions from the software usage dataset.

**Ethical Considerations**

Ethical considerations should be considered when working with this type of data. Some of these considerations include privacy, bias, use of data, and transparency. It is important to ensure that the data being collected is done in a way that respects the privacy of the individuals involved. This can be achieved by ensuring that sensitive information is kept confidential and by obtaining informed consent from participants. Also, there is a risk of bias in the collection and analysis of data, which can lead to unfair or inaccurate results. Steps should be taken to minimize bias by using standardized data collection methods, ensuring that all the simulation runs are representative, and analyzing the data objectively (O'Leary, 2021, p. 74). Next, it is important to consider how the data will be used and who will have access to it. An analyst should use the data only for the purposes for which it was collected and ensure that it is not used to discriminate against individuals or groups. Finally, being transparent about the methods used to collect and analyze the data is important. Researchers should be clear about the data they are collecting, how it will be used, and who will have access to it. As a result, ethical considerations should be taken seriously when working with any type of data, as the potential impact on individuals and society can be significant.

Before delving into the findings derived from the data analysis, it is crucial to acknowledge and address the ethical considerations that underpin the responsible handling and usage of the data. Incorporating ethical principles into the research design ensures that the insights and conclusions drawn from the data analysis are both valid and ethically sound. The following section highlights the key findings obtained from the comprehensive analysis of the dataset while keeping in mind the ethical considerations and safeguards in place to protect privacy, minimize bias, ensure appropriate data usage, and maintain transparency throughout the research process.

**Findings**

Analyzing usage patterns and identifying the most frequently utilized capabilities are crucial steps in understanding the impact of software on overall performance. The findings of this study shed light on these aspects through the examination of a comprehensive dataset and the application of statistical analysis techniques. By visualizing the data in the form of a pie chart and a bar chart, valuable insights are gained into the distribution of code usage and the prominence of specific capabilities. These visual representations provide a clear and concise overview of the patterns and trends observed within the organization. Moreover, the use of one-way ANOVA further strengthens the analysis by examining the statistical significance of the observed differences. Understanding the usage patterns and the impact of capabilities is of utmost importance as it allows organizations to optimize their software development processes, allocate resources effectively, and align their strategies with the identified trends. By delving deeper into this data analysis, valuable information can be uncovered that can lead to improved software quality, enhanced performance, and better decision-making. Therefore, exploring and interpreting these findings provides a crucial step toward maximizing the potential of software and driving organizational success in today's competitive landscape.

**Visualizing Usage Patterns for the Code**

The first research question “What are the typical usage patterns for the codes present in the dataset?” provides the hypothesis:

* H1: The code "Mirageblade" is used more frequently than the other two codes and is more likely to be successful when executed on Qualcomm Snapdragon processors than on other platforms.
* H2: There are no significant differences in the frequency and success rate of code usage across different platforms, and the three codes are used equally often in the organization.

Visualizations can help provide some initial understanding of the data. Data visualization allows the translation of complex data into a visual representation, identifies trends, and allows to display of hidden patterns if done correctly. Visualizations can allow an easier understanding of relationships between the operations and the results. Visualizations can make better business decisions and a more fluid way of having improved customer sentiment analysis. The right visualization can bring everyone to the same page, regardless of expertise level.

This is best accomplished with a visualization that communicates percentages. The clearest way for this visualization is a pie chart in percentages cluster types by average spent. This way, it would also be clear to see each cluster type and connect the percentage to an approximate size that is represented visually. Figure 1 shows that the largest cluster type is Mirageblade, followed by Phoenixstorm and Crimsonfire around equally. It is useful to understand that Mirageblade provides over 50% of the overall usage.

**Figure 1**

*Pie Chart in Percentages of Code Usage*A picture containing text, diagram, circle, screenshot

Description automatically generated

*Note.* This pie chart visually represents the percentages of code usage, providing insights into the distribution of code usage across different categories.

**Statistical Test for Typical Usage Patterns for the Code**

The purpose of a statistical test is to provide a number that describes how much the relationship between variables differs from the null hypothesis. A test can calculate the numerical probability that the difference described would be seen. This is also known as the p-value. Two basic types of variables are associated with a statistical test – quantitative and categorical. A quantitative variable shows a numerical representation, while a categorical represents a grouping.

Regression tests are looking for the cause-and-effect relationship. Linear regression is the modeling of the relationship between independent and dependent variables. The value of the dependent variable is related to the independent variable. The linear fit is the expected value (or mean) of the samples within the model. Each value of *y* has its normal distribution of where the range of *x* values may fall. Additionally, these principles can be applied to multiple linear regressions.

Finally, one-way ANOVA is a statistical technique used to compare the means of three or more groups. It examines whether the observed differences in means between the groups are significant. The analysis involves calculating the sum of squares and mean squares, which measure variation within and between the groups. The F-value and p-value are then calculated to determine if the differences in means are statistically significant. Because this research is comparing the usage of the different codes, it is appropriate to use ANOVA for this research.

Figure 2 presents an analysis of variance (ANOVA) summarizing the key findings. The Model row reveals that the independent variables accounted for some variation, with 2 degrees of freedom. The sum of squares for the model was 11,928,893, resulting in a mean square of 5,964,446. The associated F-value was 0.72, and the p-value was 0.49. Additionally, the R-Squared value was 0.000143, indicating that the independent variables explain a very small portion of the variance in the dependent variable. The Error row highlights that there were 9,997 degrees of freedom for the unexplained or residual variation. The sum of squares for the error was 83,321,403,607, with a mean square of 8,334,641. Furthermore, the Coeff Var (Coefficient of Variation) was 57.73, indicating relatively high variability in the data compared to the mean. The Root MSE (Root Mean Square Error) was 2886.98, representing the average prediction error of the model. Lastly, the Corrected Total row indicates 9,999 degrees of freedom for the total variation, with a sum of squares of 83,333,332,500. In conclusion, while the model captured some variation, the F-value and p-value suggest that its explanatory power was not statistically significant. Most of the variation remained unexplained, as evident from the substantial sum of squares for error. The R-Squared value confirms that the independent variables explain only a trivial amount of the dependent variable's variance.

**Figure 2**

*One-Way ANOVA Summary for Typical Usage Patterns for the Code***A screenshot of a computer

Description automatically generated with medium confidence**

*Note.* Using One Way ANOVA, this summary presents an overview of typical usage patterns for the code, allowing for comparisons and analysis of variations in code usage.

In Figure 3 the provided analysis includes two tests: Levene's Test for Homogeneity of identification Variance and Welch's ANOVA. Levene's test examines the equality of variances among different levels of the code factor. The results show that the code does not significantly affect the identification variance, as indicated by the non-significant F-value and p-value. Welch's ANOVA also compares the means of the code factor levels. The analysis suggests no significant differences in the means, as the F-value and p-value are not statistically significant. Additionally, the subsequent presentation of means and standard deviations for each level further supports the lack of substantial variation among the code levels.

Additionally, Figure 3 shows the Least squares means which provides adjusted means for the code levels. The Tukey-Kramer adjustment method was employed for multiple comparisons. The adjusted means indicate that there are no significant differences among the levels of the code, as the pairwise comparisons yield non-significant p-values. Overall, the analyses and comparisons consistently suggest that the code does not have a statistically significant impact on the identification variable. Both Levene's Test and Welch's ANOVA demonstrate that the code does not significantly influence the variation and means of the identification variable. The subsequent analyses and pairwise comparisons support this finding, indicating no significant differences among the code levels in terms of identification.

**Figure 3**

*One-Way ANOVA Details for Typical Usage Patterns for the Code*A screenshot of a computer

Description automatically generated with medium confidence *Note.* Building on the previous figure, this detailed analysis using one-way ANOVA provides further insights into the specific characteristics and variations of typical usage patterns for the code.

Next, Figure 4 shows the Least Squares Means (LSMEANs) difference plot which is a visual representation used in ANOVA to illustrate the pairwise differences between least squares means for different levels of a factor. It allows for easy interpretation of the magnitude and direction of the mean differences, providing insights into significant contrasts among the factor levels. It describes 5,060 for the adjusted least squares mean on Crimsonfire, just under 5,000 adjusted least squares for Mirageblade, and around 4,960 least squares for Phoenixstorm. These adjusted least squares provide estimates of the average values of id for each level of the code, after accounting for other factors in the analysis. Based on these values, it appears that there may be some variation in identification across the codes.

**Figure 4**

LSMEANs *for Code Usage*

**A screenshot of a graph

Description automatically generated with medium confidence**

*Note.* This figure showcases the least squares means, providing a comprehensive view of code usage across different factors and levels.

Finally, Figure 5 provides the Tukey-Kramer adjustment which accounts for multiple comparisons when conducting pairwise comparisons among group means. It helps control the overall Type I error rate by adjusting the significance levels, ensuring that the differences between group means are tested appropriately while considering the increased probability of making false discoveries due to multiple comparisons. Figure 5 demonstrates that there is no statistical significance for each code. The analysis demonstrates conclusively that there is no statistical significance, and the hypothesis should be rejected. This demonstrates that there are no significant differences in the code usage across different platforms, and the three codes are used equally often in the organization.

**Figure 5**

*Comparison of Code by Identification***A picture containing text, line, diagram, plot

Description automatically generated**

*Note.* This visual comparison presents a comprehensive analysis of the code usage patterns based on different identification factors, facilitating a deeper understanding of variations in code usage.

**Visualizing for Frequency of Capabilities**

For the second research question, “Which capabilities are being utilized most frequently within the organization, and what is their impact on overall performance?” the hypothesis is:

* H1: The capability of "Nonlinear analysis" is being utilized most frequently within the organization and is positively impacting overall performance.
* H2: There are no significant differences in the frequency of utilization and impact on overall performance among different capabilities within the organization.

The bar chart in Figure 6 presents the frequency of capabilities utilized within our organization, aiming to shed light on their impact on overall performance. Analyzing the frequency of capabilities can help identify which capabilities are being utilized most frequently and gain insights into their potential influence on our organization's performance. The chart provides a visual representation of the frequency distribution of capabilities used within Elemental Software Solution. This analysis allows the exploration of the relationship between capability utilization and overall performance, helping to understand which capabilities contribute significantly to the success of the organization.

**Figure 6**

*Bar Chart of Frequency of Capabilities***A picture containing text, screenshot, number, diagram

Description automatically generated**

*Note.* Represented as a bar chart, this figure illustrates the frequency distribution of capabilities, offering insights into the prevalence and occurrence of different capabilities.

**Statistical Test for Frequency of Capabilities**

Figure 7 explores the relationship between the dependent wall time and the categorical capability use. The model, with 13 degrees of freedom, indicates that it explains only a small portion of the variability in wall time, as reflected by the low R-square value of 0.000793. The associated F-value of 0.61 and p-value of 0.85 suggest that the model's explanatory power is not statistically significant, implying that the observed differences in wall time may be due to random chance rather than meaningful factors related to capability use. Additionally, the larger sum of squares for error highlights that most of the variability in wall time remains unexplained by the model, indicating the presence of other factors that may have a stronger influence on wall time. As a result, the data does not provide strong evidence of a significant relationship between capability use and wall time, suggesting the need to consider additional factors that could contribute to the observed variations.

**Figure 7**

*One-Way ANOVA Summary for Frequency of Capabilities*A screenshot of a computer

Description automatically generated with medium confidence

*Note.* Using One-Way ANOVA, this summary highlights key information about the frequency of capabilities, allowing for comparisons and analysis of variations in capability occurrence.

The data includes information on the wall time and its relationship with the capability use. The analysis begins with Levene's Test for homogeneity of wall time variance, which assesses the equality of variances across different capability uses. The resulting F-value of 0.89 and p-value of 0.57 indicate that there is no significant evidence to suggest differences in variances between capability uses. Next, the ANOVA of squared deviations from group means is conducted. The analysis reveals that the model's explanatory power, represented by the F-value of 0.61 and p-value of 0.85, is not statistically significant, suggesting that the observed differences in wall time among capability used may be due to chance rather than meaningful factors.

The subsequent table presents the means and standard deviations of wall time for each capability used. It shows the average wall time and the associated variability within each capability use category. Lastly, LSMEANs and pairwise comparisons among the capability uses are provided. The LSMEANs represent the adjusted means accounting for multiple comparisons. The table presents the LSMEAN values and their corresponding numbers for each capability use category. The subsequent table presents the p-values for pairwise comparisons between capability uses, indicating whether the differences between the LSMEANs are statistically significant. As a result, the data does not provide strong evidence of significant capability use and wall time. The statistical analyses and pairwise comparisons do not demonstrate any statistically significant differences among the capability uses in terms of wall time.

**Figure 8**

*One-Way ANOVA Details for Frequency of Capabilities*A screenshot of a computer

Description automatically generated with low confidence

*Note.* Building on the previous figure, this detailed analysis provides a comprehensive examination of the frequency of capabilities, enabling a deeper understanding of specific patterns and variations.

Post-processing, positioned slightly below 51, indicates a relatively higher LSMEANs value for that capability, suggesting it has a potentially significant impact on the dependent variable. Mesh generation, slightly above 50, also shows a relatively higher LSMEANs value. Element library, precisely at 50.5, represents an average LSMEANs value for that capability. On the other hand, both multiphase flow modeling and multiphysics capabilities cluster around 47, indicating relatively lower LSMEANs values. These findings suggest that post-processing and mesh generation capabilities may have a more significant impact on the dependent variable compared to multiphase flow modeling and multiphysics capabilities. The scatter plot allows for visual comparison and understanding of the LSMEANs values, aiding in identifying the capabilities with potentially greater influence on the outcome.

**Figure 9**

LSMEANs *for Frequency of Capabilities*A picture containing text, screenshot, number, font

Description automatically generated

*Note.* This figure presents the least squares means, providing a comprehensive view of capability frequency across different factors and levels.

Based on the Tukey-Kramer adjustment results, all the lines connecting the different capabilities in the plot are labeled as not significant. This implies that there are no statistically significant differences observed between the LSMEANs of the capabilities when considering the adjusted significance levels. The Tukey-Kramer adjustment accounts for multiple comparisons and helps control the overall Type I error rate, ensuring that differences between group means are appropriately tested while considering the increased probability of false discoveries. Therefore, the non-significant labels suggest that the LSMEANs for the capabilities do not differ significantly from each other, indicating similar impacts on the dependent variable across the different capabilities.

**Figure 10**

*Comparison of Frequency of Capabilities*A picture containing text, screenshot, line, parallel

Description automatically generated

*Note.* Through visual comparison, this figure offers insights into the variations and relationships between different capabilities based on their frequency.

**Forecasting**

Forecasting data plays a crucial role in various aspects of business and decision-making. One of the key benefits of forecasting is its ability to inform and guide decision-making processes. By analyzing historical data and identifying trends and patterns, forecasting provides valuable insights into future outcomes. This enables organizations to make informed decisions regarding production levels, inventory management, resource allocation, and strategic planning. By anticipating changes in demand, market conditions, and other factors, businesses can proactively adapt their strategies to meet future requirements and stay ahead of the competition. Furthermore, forecasting helps in identifying potential risks and opportunities, allowing organizations to mitigate risks and capitalize on emerging trends. Overall, forecasting empowers businesses with the knowledge and foresight needed to optimize operations, improve efficiency, and achieve their goals. ARIMA 3-1-1 was used to analyze the data. ARIMA 3-1-1 is a mathematical model used to analyze and predict time series data. It considers the patterns and changes in the data over time by considering the relationship between past observations, the differences between consecutive observations, and the average of past errors.

Figure 11 presents the time forecasting analysis for the wall time. The series was differenced once, and the mean of the working series is approximately -0.00192. The standard deviation is 39.75, indicating moderate variability. The autocorrelation check reveals significant autocorrelation at multiple lag intervals, with coefficients ranging from approximately -0.49 to 0.02. These findings suggest a slight downward trend in the data and the presence of autocorrelation, indicating a departure from a white noise process.

**Figure 11**

*Overview of Modeling and Forecasting for Wall Time*A picture containing text, screenshot, font, number

Description automatically generated

*Note.* This figure provides an overview of the modeling and forecasting process for wall time, presenting key components and stages involved in analyzing and predicting wall time behavior.

Figure 12 presents valuable insights into the time series data. The scatter plot showcases the data points scattered across the range of -100 to 100, indicating variability within the 10,000 observations. The Autocorrelation Function (ACF) plot reveals a strong relationship between consecutive observations, with a diminishing autocorrelation over time. The Partial Autocorrelation Function (PACF) table shows significant negative partial autocorrelation coefficients, indicating an inverse relationship between observations at specific lags. These visualizations provide important information about patterns and dependencies in the time series data.

**Figure 12***Trend and Correlation Analysis for Wall Time*A picture containing text, screenshot, plot, line

Description automatically generated

*Note.* This figure showcases the analysis of trends and correlations in wall time data, enabling the identification of patterns, relationships, and dependencies within the time series.

Figure 13 showcases the ARIMA Estimation Optimization and Residual Analysis. The estimation process utilized the maximum likelihood method to estimate five parameters, with convergence evaluated based on the maximum relative change in estimates. The resulting estimation involved nine iterations, although a warning message suggests a possible lack of full convergence. Parameter estimates, along with their standard errors, t-values, and p-values, are provided for the ARIMA model. The AIC and SBC values indicate model fit, and the correlation matrix illustrates the relationships between the estimated parameters. Additionally, the autocorrelation chec­k assesses residual autocorrelation at various lags, aiding in evaluating the presence of remaining autocorrelation in the model's residuals.

**Figure 13**

ARIMA Estimation Optimization and Residual Analysis *of Modeling and Forecasting for Wall Time*A screenshot of a computer

Description automatically generated with medium confidence

*Note.* Building on the previous figure, this analysis focuses on the ARIMA estimation optimization process and residual analysis, providing insights into the estimation methodology, convergence, and goodness-of-fit assessment.

Figure 14 presents four charts: ACF, CACF, IACF, and White Noise Probability. The ACF chart shows a strong autocorrelation at lag 0 with 1.0, while other lags have minimal positive values. The PACF chart has noise around 0, making it challenging to determine its polarity. The IACF chart mirrors the ACF pattern. The White Noise Probability displays decreasing values beyond lag 15, indicating a diminishing influence of past observations. These charts provide insights into the autocorrelation, noise, and randomness of the time series data.

**Figure 14**

*Residual Correlation Diagnostics for Wall Time*

A picture containing text, diagram, line, number

Description automatically generated *Note.* This figure presents diagnostics for residual correlation in wall time data, helping to evaluate the presence of autocorrelation and identifying any remaining dependencies in the residuals.

In Figure 15, the residual normality diagnostics provide insights into the distribution of residuals, indicating whether they follow a normal distribution. The chart reveals that most of the residuals exhibit a normal distribution, except for deviations observed in the central and upper regions. Furthermore, the QQ-Plot displays departures from the expected line in the tails, suggesting deviations from normality. These residual diagnostics help assess the assumption of normality and identify potential areas of concern in the residual distribution.

**Figure 15***Residual Normality Diagnostics for Wall Time*A picture containing plot, line, diagram, screenshot

Description automatically generated

*Note.* This figure focuses on assessing the normality of residuals in wall time data, examining their distribution and comparing them to the expected normal distribution using techniques such as QQ-plots.

Figure 16 provides crucial information regarding the model for wall time and its forecasts. The estimated mean for the variable is 0.00017, representing the average expected value. The model incorporates autoregressive and moving average factors, which are characterized by specific coefficients. These factors indicate the relationship between past and current observations, enabling the analysis of time series behavior. Additionally, the forecasts table presents predicted values for wall time, along with their corresponding standard errors and 95% confidence limits. These forecasts play a vital role in resource planning, scheduling, and process optimization that rely on time-related factors. The data offers valuable insights into wall time, empowering analysts and decision-makers to understand trends, patterns, and make informed predictions. Also, Figure 17 shows a forecast between 0 and 100 with the predicted being 50 based on the previous data.

**Figure 16**

*Model Estimation and Wall Time Forecasts*A screenshot of a computer

Description automatically generated with low confidence

*Note.* This figure highlights the estimation of the wall time model and presents forecasts, providing valuable information for resource planning, scheduling, and process optimization that rely on accurate time-related.

**Figure 17**

*Forecasting for Wall Time*A picture containing text, screenshot, rectangle, line

Description automatically generated

*Note.* This figure specifically focuses on the forecasting aspect of wall time, presenting predicted values and their corresponding intervals to assist in understanding future trends and making informed decisions.

Figure 18 presents the results of outlier detection, providing valuable insights into the presence of anomalous observations in the dataset. The outlier detection summary reveals that a maximum of five observations were searched for outliers, out of which two outliers were found. A significance level of 0.05 was used to determine the presence of outliers, indicating a level of confidence in the detected anomalies. The outlier details table specifies the observations identified as outliers, their types (e.g., shift), estimated values, and corresponding chi-square statistics with approximate probability values. Specifically, observation 9198 is flagged as a shift outlier with an estimated value of -2.83 and a chi-square statistic of 4.22 with an approximate probability of 0.04. Similarly, observation 4 is also identified as a shift outlier with a significantly larger estimated value of -42.60 and a chi-square statistic of 4.03 with an approximate probability of 0.04. These outlier detection results highlight specific data points that deviate significantly from the expected patterns, allowing further investigation into potential anomalies, data quality issues, or other factors that may influence the analysis.

**Figure 18**

*Outlier Detection Summary for Wall Time*

A picture containing text, screenshot, font, number

Description automatically generated

*Note.* This figure summarizes the detection of outliers in the wall time data, providing information on the number of outliers found, their significance, and the detection methodology employed.

**Dashboard Code Usage**

Visualizing data on a dashboard is important beyond the analysis itself as it provides a comprehensive view of the data, allowing for the identification of patterns, trends, and outliers. It simplifies complex information, making it accessible to a broader audience and facilitating faster decision-making. Dashboards offer interactivity, enabling users to explore the data further and gain deeper insights. Real-time updates ensure that the information displayed remains current, aiding in monitoring and tracking key metrics. Overall, dashboards enhance data comprehension, communication, collaboration, and provide timely insights, making them a valuable tool for decision-making and gaining insights from complex data.

The dashboard looks at the views that impact the overall strategy of Elemental Software Systems. The dashboard shows that there are a total of 453 users who mostly use release, but also are prone to using VOTD. An actionable step would be working with the analysts to encourage using the more stable release version of the code, and less use of VOTD. This shows the organization that it would be critical to ensure that the capabilities with nonlinear analysis, the element library, and post-processing would be fully robust. In some situations, it would also be helpful to understand if a capability was not being used, and therefore it could be deprecated. The users per month appear mostly stable but moving downward slightly. This would be good to be investigated to understand if this is just temporary or if there is a driver that the number of users is decreasing. Next, the platforms by wall time show that Arduino and Intel NUC are the most heavily used, followed by Tinker Board. It would be critical to have conversations to ensure those platforms were properly maintained to avoid downtime and seek future investment in similar platforms. Finally, the version number shows that most runs that are being done overall are from the version of the day and there are many failures. This would want to be investigated to understand the reasons why failures were happening to see if they can be avoided. This dashboard gives the option to select this information for the specific code and over a specific date to understand details that might have been changed in the code that the developer community may be interested in tracking.

**Figure 19**

*Dashboard of Elemental Systems Code Usage***A close-up of a chart

Description automatically generated with low confidence**

*Note.* This comprehensive dashboard provides an overview of Elemental Software Systems code usage, consolidating key information and visualizations to facilitate data exploration and analysis.

**Conclusion**

In conclusion, the analysis conducted reveals important insights into the utilization of capabilities and its impact on overall performance within our organization. The findings indicate that there are no significant differences in code usage across different platforms. This suggests that each code is being utilized equally frequently, highlighting a consistent pattern of code utilization within Element Software Systems. However, when examining the relationship between wall time and categorical capability use, the analysis indicates that the model's explanatory power is not statistically significant. This implies that the observed differences in wall time may not be directly attributed to the utilization of specific capabilities. Instead, these differences might be influenced by other factors not considered in the analysis or may even occur due to random chance.

It is important to acknowledge that the scope of our analysis was limited to the relationship between categorical capability use and wall time. Thus, to gain a more comprehensive understanding of the factors influencing wall time and their impact on overall performance, future research should incorporate a broader range of variables and factors into the analysis. By expanding the scope of the investigation to encompass these additional factors, the data can uncover a more nuanced understanding of the dynamics between capability utilization, wall time, and overall performance. This deeper analysis will provide valuable insights into the factors that truly drive performance and productivity, enabling Elemental Software Solutions to develop more effective strategies and recommendations for optimizing our organizational processes.

**Recommendations**

In addition to exploring variables such as system configuration, user behavior, and external influences, future research should also delve into other factors that may influence wall time determinants. These could include factors such as development team composition, project complexity, software architecture, and the availability of development resources. By incorporating a broader range of variables into the analysis, we can gain a more comprehensive understanding of the various factors that impact wall time and overall performance. This expanded approach will enable researchers to develop more robust predictive models that provide deeper insights into the factors influencing software development timelines.

By developing more advanced predictive models that consider multiple factors, organizations can optimize their resource planning processes. This would enable them to allocate resources more effectively, identify potential bottlenecks in the development process, and streamline their workflows. A comprehensive understanding of wall time determinants, derived from data-driven analysis, will allow organizations to make data-informed decisions and optimize their software development strategies accordingly. This proactive approach to resource planning and optimization can lead to improved productivity, cost savings, and higher customer satisfaction.

The dataset of software usage holds immense potential for organizations operating in the software development domain. By leveraging this dataset and embracing data-driven decision-making, organizations can make informed choices throughout the software development lifecycle. Optimizing resource allocation based on insights from the dataset can result in improved software quality, reduced defects, enhanced customer satisfaction, and an improved organizational reputation. Additionally, analyzing the dataset can provide organizations with valuable insights into user behavior, allowing them to tailor their software products to meet user needs and preferences more effectively.

Moreover, gaining insights into user behavior from the dataset can enable organizations to enhance the user experience of their software products. This, in turn, can lead to increased user engagement, greater customer loyalty, and higher customer retention rates. By utilizing data-driven analysis of the dataset, organizations can identify opportunities for user experience enhancement and tailor their software development efforts accordingly. This user-centric approach to software development can help organizations differentiate themselves from competitors and build strong, long-lasting relationships with their user base.

Furthermore, optimizing development workflows based on data-driven analysis of the dataset can have a profound impact on development efficiency. By identifying inefficiencies, bottlenecks, and areas of improvement within the development process, organizations can reduce time-to-market for their software products and increase overall team productivity. This optimization of development workflows can significantly contribute to achieving organizational goals, delivering high-quality software products, and staying competitive in the dynamic landscape of software development. By continuously analyzing the dataset and refining their development processes, organizations can drive continuous improvement and stay at the forefront of the industry.

In conclusion, the power of leveraging the dataset on software users extends far beyond the realm of research. It has real-world implications for organizations aiming to excel in software development. By embracing data-driven decision-making, organizations can enhance their software development processes, make informed choices, and achieve their software development goals. The dataset offers valuable insights that can drive improvements in software quality, user experience, and development efficiency. Transitioning from the current state to this better future state requires a change in mindset and a commitment to embracing the potential offered by the dataset. By harnessing its power, organizations can thrive in the ever-evolving world of software development and position themselves as leaders in their respective domains (Nelson, 2018, p. 267).

# References

Andrei, O., & Calder, M. (2018). Data-driven modelling and probabilistic analysis of interactive software usage. *Journal of Logical and Algebraic Methods in Programming, 100*, 195-214. Retrieved from https://www.sciencedirect.com/science/article/pii/S2352220818300178

Biesialska, K., Franch, X., & Muntés-Mulero, V. (2021, 4). Big Data Analytics in Agile software development: A systematic mapping study. *Information and Software Technology, 132*. Retrieved from https://csuglobal.idm.oclc.org/login?url=https://search.ebscohost.com/login.aspx?direct=true&db=edselp&AN=S0950584920301981&site=eds-live

Deng, Q., Wellsandt, S., Hribernik, K., & Thoben, K.-D. (2021). Understanding Users and Products in Product Development: The Application of Product Usage Information and its Challenges. *Proceedings of the Design Society, 1*, 3299-3308. Retrieved from https://www.cambridge.org/core/article/understanding-users-and-products-in-product-development-the-application-of-product-usage-information-and-its-challenges/EA03C9DA97CCE793337EEFE8CF74B802

Gupta, M., George, J. F., & Xia, W. (2019, February). Relationships between IT department culture and agile software development practices: An empirical investigation. *International Journal of Information Management, 44*, 13-25. doi:10.1016/j.ijinfomgt.2018.09.006

McNulty, R. (2022, 11). A logical analysis of null hypothesis significance testing using popular terminology. *BMC Medical Research Methodology, 22*(1), 1-9. Retrieved from https://csuglobal.idm.oclc.org/login?url=https://search.ebscohost.com/login.aspx?direct=true&db=aph&AN=160307167&site=eds-live

Meyer, M., Wiederkehr, I., Koldewey, C., & Dumitrescu, R. (2021). Understanding Usage Data-Driven Product Planning: A Systematic Literature Review. *Proceedings of the Design Society, 1*, 3289-3298. Retrieved from https://www.cambridge.org/core/article/understanding-usage-datadriven-product-planning-a-systematic-literature-review/14E6976F78E36A42AB487B1F92768F2B

Nelson, G. S. (2018). *The Analytics Lifecycle Toolkit* (1st ed.). Hoboken, NJ: John Wiley & Sons, Inc.

O'Leary, Z. (2021). *The Essential Guide to Doing Your Research* (4th ed.). (J. Seaman, Ed.) Thousand Oaks, CA: SAGE Publications Ltd.

Santos Júnior, P., Barcellos, M., Falbo, R., & Almeida, J. (2021, 8). From a Scrum Reference Ontology to the Integration of Applications for Data-Driven Software Development. *Information and Software Technology, 136*. Retrieved from https://csuglobal.idm.oclc.org/login?url=https://search.ebscohost.com/login.aspx?direct=true&db=edselp&AN=S0950584921000537&site=eds-live

Unahalekhaka, A., & Bers, M. (2021, 6). Taking coding home: analysis of ScratchJr usage in home and school settings. *Educational Technology Research & Development, 69*(3), 1579-1598. Retrieved from https://csuglobal.idm.oclc.org/login?url=https://search.ebscohost.com/login.aspx?direct=true&db=pbh&AN=151271348&site=eds-live

**Appendix**

**Appendix A**

|  |  |  |  |
| --- | --- | --- | --- |
| Variable | Data Type | Values | Description |
| id | numeric | 1 - 10000 | A unique identifier for each data point. |
| Date | date | 1/1/2019 – 4/13/2023 | The date and time when the software was run. |
| first\_name | character | Various | The first name of the user who ran the software. |
| last\_name | character | Various | The last name of the user who ran the software. |
| username | character | Various | The username of the user who ran the software. |
| Code | character | Crimsonfire, Mirageblade, Phoenixstorm | The name of the code or module that was run. |
| Platform | character | Arduino, BeagleBone Black,  HiKey 960, Intel NUC, NVIDIA Jetson, ODROID, Qualcomm Snapdragon, Tinker Board | The platform or device on which the code was run. |
| Organization | numeric | 1 - 9990 | The organization or company to which the user belongs. |
| Processors | numeric | 1 - 3199 | The number of processors or CPU cores used during the run. |
| Wall Time | numeric | 0.10 – 99.00 | The total time taken for the run to complete. |
| Status | character | Success, Failure | The status of the run, either Success or Failure. |
| Version | character | EndOfSprint, Release, VOTD | The version of the software used. |
| Version Number | character | VOTD, 4.0, 4.1, 4.2, 4.3, 4.4, 4.5, 4.6, 4.7, 4.8, 4.9, 5.0, 5.1, 5.2, 5.3, 5.4, 5.5, 5.6, 5.7, 5.8, 5.9, 6.0 | The version number of the software used. |
| AverageMemoryUse | numeric | 8 - 115 | The average amount of memory used during the run. |
| Capability Use | character | Boundary conditions, Combustion modeling, Element library, Heat transfer modeling, Material modeling,  Mesh generation, Multiphase flow modeling, Multiphysics capabilities, Nonlinear analysis, Optimization,  Post-processing, Radiation modeling, Solver, Turbulence modeling | The specific capability or function of the software that was used during the run. |