CISC 322

A3

Kodi - Architectural Enhancement

Date: December 5, 2023

Authors:

Ricardo Chen (Team Leader) – 21rc20@queensu.ca
ZhouJun Pan (Presenter) – 19zp9@queensu.ca
Jiahao Ni (Presenter) – 19jn35@queensu.ca
Lucy Zhang – 19ytz@queensu.ca
Shuaiyu Chen - 21sc33@queensu.ca

1.0 Abstract

This report proposes a new enhancement for the Kodi media player app – the integration of a comprehensive "Add-on Reviews and Ratings" feature. Currently absent in the Kodi ecosystem, this functionality aims to empower users with informed decision-making when selecting and installing add-ons. The report delves into the rationale and practicality of this enhancement, detailing its implementation through two compelling use cases illustrated by sequence diagrams. The effects of this enhancement are multifaceted, contributing to a more engaged and informed Kodi community. Users can now share their experiences, providing insights into the performance and reliability of add-ons. Potential risks and limitations are thoughtfully considered, including concerns about database consistency, server load, and the need for a robust moderation system to maintain content integrity. By acknowledging these challenges, the report aims to facilitate a thorough understanding of the potential hurdles associated with implementing "Add-on Reviews and Ratings." The proposed enhancement not only addresses a crucial functionality gap but also sets the stage for a more collaborative and user-friendly Kodi experience. The report provides a comprehensive analysis of the enhancement.

2.0 Introduction and Overview

In the world of media consumption, Kodi has emerged as a versatile platform offering a large number of add-ons to enrich user experiences. However, the absence of an "Add-on Reviews and Ratings" feature limits users' ability to make informed choices when selecting add-ons. This enhancement is motivated by the practice in app stores, where users heavily rely on reviews and ratings to foresee the reliability and functionality of an application before downloading. Much like the app store experience, users engaging with Kodi seek to optimize their media playback environment by selecting add-ons that align with their preferences and needs. The motivation behind this enhancement is to provide Kodi users with a similar evaluative tool. By introducing reviews and ratings, users can make informed decisions, avoiding potential frustration with poorly performing or unreliable add-ons. The primary goal is to enhance the user experience by empowering the community to share their insights. Users can contribute by providing feedback on add-ons, creating a collaborative ecosystem where the collective wisdom of the community aids in the selection process. This user-centric approach not only benefits those seeking add-ons but also encourages developers to refine and improve their offerings based on user feedback. This report will delve into the Functional and Non-Functional requirements, conduct a SAAM analysis for alternatives, assess the influence on high- and low-level conceptual architecture, and illustrate the proposed enhancement through two use cases and sequence diagrams. The motivation behind this enhancement is to elevate the Kodi user experience by promoting informed decision-making, fostering community collaboration, and ensuring users find add-ons that align with their expectations.

3.0 Proposed Enhancement

We propose to add an enhancement to the Kodi system with a rating and review system for add-ons.

Users' lack of direct knowledge of add-on quality and performance is a significant challenge when selecting and installing add-ons. Often, users have to rely on third-party websites to find and obtain information about add-ons, a process that is time-consuming and the results are not always reliable. Currently, Kodi allows users to download and install various add-ons from official repositories and third-party sources. Still, there is no built-in functionality within the system to directly collect and display user feedback on these add-ons. Users have no direct way to judge the quality or reliability of an add-on beyond download counts and basic descriptions. To understand the performance and reliability of an add-on, it is often necessary to rely on the Kodi community forums or other resources on the Internet. This method of obtaining information is not only inconvenient but also difficult to ensure the timeliness and accuracy of the information. Especially for new users, this may lead to confusion and uncertainty when choosing add-ons. In addition, this dispersed source of information also limits the possibility of effective communication within the community, making it difficult for users' feedback and experiences to be obtained and utilized promptly by other users. Therefore, the current Kodi system has obvious gaps in add-on evaluation and user feedback, which not only affects the user experience but also limits the development of the community and add-on system.

However, implementing a built-in rating and review system will directly change this situation, allowing users to obtain more comprehensive information based on the evaluations of others when making decisions, thereby improving the quality of their choices. Also, a key advantage of Kodi is its active user community. Introducing this feature, will not only encourage users to share their experiences and insights and increase interaction within the community but also provide feedback to add-on developers, which can also promote and contribute to the continuous improvement of add-ons. In addition, this quality-checking mechanism will help highlight better add-ons, while helping to identify and avoid the use of low-quality or problematic add-ons. This will improve the quality and security of the entire add-on system. In short, the built-in rating and review system will simplify the add-on selection process for new users, it will also quickly guide them to community-recommended add-ons, which will greatly enhance the overall user experience of Kodi.

Next, we will show the relationship between the new add-on rating and review system and several key functions. First, it will interact directly with the add-on management system, allowing users to see ratings and reviews from other users when browsing or searching for add-ons. Second, this system may need to be integrated with the user account system, especially when users submit ratings and comments. This integration will ensure the authenticity and reliability of feedback while helping maintain order in the community. Additionally, to display ratings and collect user reviews, Kodi’s user interface (UI) will need to be adjusted accordingly. This may include adding a rating display and comment areas to the add-on details page. Such adjustments are designed to keep the user interface intuitive and user-friendly while effectively providing the required information. Finally, the add-on's metadata structure will also undergo adjustments to include ratings and review data, which will be used for sorting and search functionality within the library.

4.0 The Effects of the Enhancement

The "Add-on Reviews and Ratings" enhancement in Kodi seamlessly interacts with existing features, enhancing the overall user experience. It is designed to seamlessly integrate into the existing Kodi architecture without significant disruptions of the current functionalities. However, the introduction of the "Add-on Reviews and Ratings" enhancement in Kodi significantly influences various aspects of the system's architecture and functionality.

In the context of Kodi's existing concrete architecture, characterized by the Publish/Subscribe (Pub-Sub) style, the proposed "Add-on Reviews and Ratings" enhancement aligns seamlessly with the modular and flexible nature of the system. The Pub-Sub style facilitates communication between components, allowing for independent interaction and dynamic addition of new features.

Maintainability: High

The "Add-on Reviews and Ratings" enhancement introduces a modular and self-contained structure, aligning well with Kodi's overall architecture. Components related to reviews and ratings can be maintained independently, minimizing disruptions to the broader system. This modular design enhances maintainability by allowing targeted updates and modifications without extensive dependencies on existing Kodi components.

Evolvability: Medium

While the enhancement integrates seamlessly with the existing system, allowing for incremental additions of new features, there are considerations regarding potential dependencies and the impact on the overall system's evolution. The decentralized nature, while beneficial for maintainability, may introduce challenges in coordinating system-wide changes, warranting a more cautious assessment of evolvability.

Testability: Medium

While the introduction of new testable components is a positive aspect, the evaluation of the enhancement's testability is medium. Testing scenarios can assess the accuracy of review data presentation and user interactions, but challenges may arise in comprehensive testing due to the decentralized nature of the Pub-Sub architecture. Specific attention to end-to-end testing and scenario-based testing will be crucial for thorough evaluation.

Performance: Medium

The performance of the "Add-on Reviews and Ratings" enhancement is rated as medium. Database interactions for storing and retrieving reviews may introduce considerations for optimal performance. The rigorous load testing protocols are necessary to assess the impact on system responsiveness, particularly during peak user engagement. Fine-tuning database interactions and optimizing query performance will be essential to elevate the rating to high.

5.0 SAAM Analysis

5.1 Approach 1 – Kodi API Integration

The primary goal of Approach 1 is the integration of Kodi's Ratings and Reviews functionality into the Kodi app. This involves steps such as developer account creation, API key generation, codebase modification, and user interface (UI) design. Key stakeholders include developers responsible for the integration and users who contribute and consume reviews.

Developers start by creating an account on the Kodi Developer Portal to access resources for plugin development. The generated API key from the Developer Portal serves as a secure identifier during communication with Kodi's Ratings and Reviews API. This API key is embedded into the Python-based plugin codebase, facilitating secure interactions with the external Kodi API.

A dedicated section in the UI is crafted to dynamically display reviews and ratings retrieved from the Kodi API. This section enhances user engagement by providing a clear and visually appealing interface. Additionally, a review submission feature is implemented, allowing users to submit reviews and ratings. Before submission, users undergo a robust authentication process, either logging into their Kodi accounts or using alternative authorization methods.

Approach 1 is centered around Kodi API integration and provides a solid foundation for introducing Ratings and Reviews. It utilizes the official Kodi API for storing and retrieving review data. However, it relies on the existing Kodi infrastructure and API endpoints. The complexity of API integration and codebase modification may pose challenges, particularly for less experienced developers. The approach's dependency on the stability and reliability of Kodi's Ratings and Reviews API is a potential drawback.

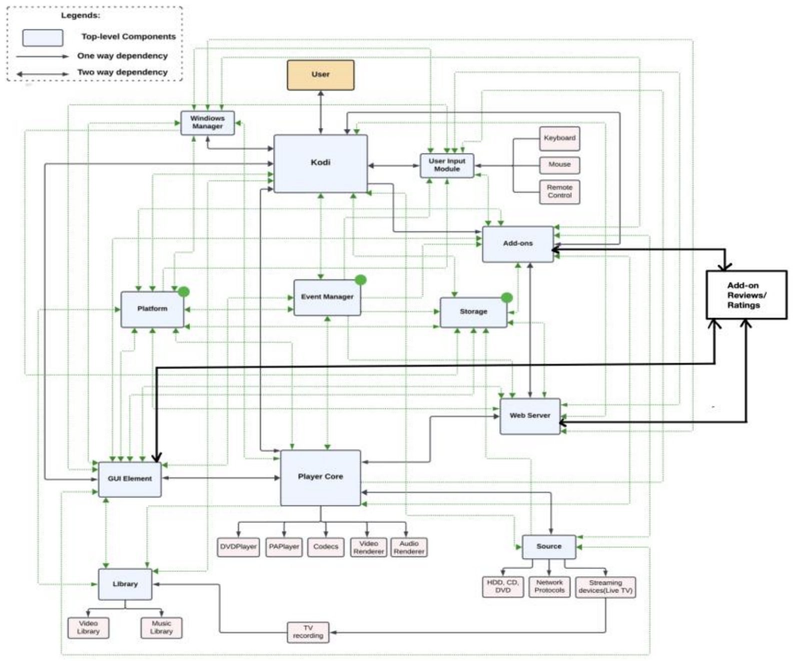

Figure 1: Updated Conceptual Illustration of Approach 1 for Add-on Reviews and Ratings

5.2 Approach 2 - Custom Ratings and Reviews System

Approach 2 introduces a personalized Ratings and Reviews system for Kodi media player plugins, transforming user engagement. This method involves creating a dedicated database table, implementing API endpoints for seamless communication, and designing a user-friendly frontend using Vue.js or React. The database table, fortified with SQLite or MySQL, ensures smooth storage and retrieval of user reviews and ratings. API endpoints are strategically bifurcated for retrieving and submitting reviews, enhancing backend-frontend communication.

The frontend, crafted with leading frameworks, offers an immersive user experience. A tailored UI component empowers users to navigate reviews easily and contribute their own. Emphasizing user-friendliness and aesthetics, this design ensures a delightful user journey. Security measures, including meticulous user authentication and tokens, safeguard the review system against potential misuse.

To maintain system integrity, asynchronous task handling addresses temporal delays in processing reviews, ensuring a responsive interface. Thoughtful feedback mechanisms assure users of successful review submission and processing. This holistic approach reflects the developer's commitment to user-centric design, incorporating advanced techniques for security, efficiency, and user satisfaction in the dynamic Kodi ecosystem. The choice of this method is contingent on plugin needs, considering user experience, security, and compatibility with the Kodi ecosystem.

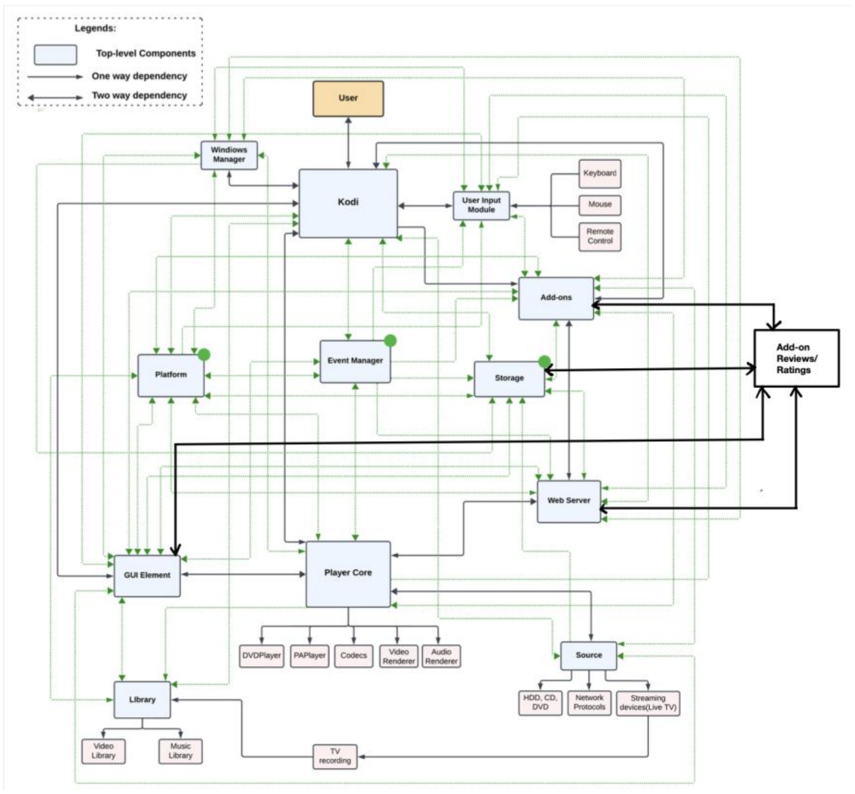

Figure 2: Updated Conceptual Illustration of Approach 2 for Add-on Reviews and Ratings

5.3 Stakeholders Analysis

User Interface Layer (UI Element)

Enhanced Decision-Making for Users: Integrating rating information within the UI allows users to make more informed choices by evaluating the quality and relevance of add-ons based on feedback from the Kodi community.

Increased Interactivity: Enabling users to rate and review directly within the interface increases user engagement and creates a feedback loop between users and developers.

Application Layer (Kodi)

Strengthened User Trust: Showcasing a commitment to user feedback through a built-in rating system reinforces trust in the Kodi platform.

Quality Control Mechanism: User feedback through ratings provides Kodi with a mechanism to monitor and ensure the quality of add-ons, maintaining a high standard for user experience.

Add-on Layer (Add-ons)

Direct Market Feedback for Developers: Developers gain direct insights from the market via user feedback, which is essential for iterating and improving their add-ons.

User Satisfaction Monitoring: Analyzing trends in user ratings and reviews can inform developers of user satisfaction levels and indicate when adjustments are needed.

Storage Layer (Storage)

Valuable Data Insights: Storing user ratings and reviews data can provide Kodi with valuable insights that aid in optimizing offerings.

Historical Feedback Record: Accumulated rating and review data serve as a historical record of user feedback, valuable for longitudinal analysis and reference.

Network Services Layer (Web Server)

Global Data Accessibility: The server layer can manage rating data from a global user base, offering real-time feedback and helping to understand user opinions promptly.

Performance Optimization Opportunities: As user rating data grows, the server layer can be optimized for performance to ensure a consistent and responsive user experience.

External Dependencies (such as Python and Web Scrapers)

Extended Functionality: External dependencies can be used to expand the functionality of the rating system, like implementing machine learning algorithms to identify fraudulent reviews.

Automated Processing: Automation tools can assist in handling and analyzing large volumes of user feedback, providing support for data-driven decision-making.

Users

Improved Choice: When deciding to download an add-on, users can preview the add-on's functionality and user reviews to decide whether they want to choose to use the add-on.

Voice and Empowerment: This system is useful because it allows users to share their experiences and influence the direction of the add-on's development through feedback.

Increased Trust: Seeing responsive developers address user concerns and improve add-ons based on user feedback increases user trust in the Kodi platform and its community of creators.

Developers

Valuable Feedback: Debugging, feature enhancements, and overall improvements to add-ons can't happen without user feedback, and developers can get direct and actionable feedback from user reviews.

Reputation Building: Positive ratings and reviews can greatly enhance a developer's reputation, leading to increased downloads and user engagement with their add-ons.

Understanding the market: Review analysis provides developers with insights into market trends, user needs and preferences, allowing them to customize their products more effectively.

Investors

Growth Potential: Adding a rating and review system increases user engagement and platform stickiness, making Kodi a more attractive investment opportunity as the user base has the potential to grow.

Improved Product Quality: As developers improve their add-ons based on user feedback, the overall quality of the Kodi ecosystem improves, which is a good trend for investors looking for long-term value.

Data-Driven Decisions: Investors can use data from reviews and ratings to make informed decisions about resource allocation, identifying areas of the Kodi ecosystem with high potential for further development and investment.

5.4 Important non-functional requirements (NFRs)

NFR	Approach 1	Approach 2
Performance	Constrained by the performance of the Kodi server, the efficiency of this method encounters certain limitations. The plugin's overall performance is predominantly contingent upon the responsiveness of the Kodi Official API. While it provides a streamlined development experience and the assurance of official support, the inherent dependence on the Kodi server introduces a degree of performance restriction.	Exhibiting a higher degree of flexibility, this method allows developers to tailor the system for optimal performance. The custom system affords greater autonomy, enabling the selection of efficient database engines, frontend frameworks, and the implementation of judicious code design. This versatility empowers developers to strategically optimize performance, catering to the specific demands of the plugin.
Scalability	Constrained by the scalability of the Kodi server, the plugin's ability to scale concurrently is subject to limitations imposed by the Kodi Official API. While this method offers simplicity and alignment with the official Kodi infrastructure, its scalability is inherently tethered to the performance and capacity of the central Kodi server.	Exhibiting a higher degree of scalability, this method empowers developers to tailor the system to meet evolving demands. Developers have the autonomy to choose a database engine that aligns with specific requirements and can implement horizontal or vertical scaling strategies to accommodate a growing user base and increasing data volumes. This enhanced scalability ensures that the custom system remains responsive and adaptable in the face of expanding user and data needs.
Testability	The official API typically comes equipped with corresponding testing tools and documentation, rendering the testing process straightforward. However, testing the specific business logic within the plugin may be relatively constrained. While it assures ease of testing on the API level, testing intricate functionalities within the	The custom system allows for a more flexible design of testable code structures, facilitating comprehensive unit testing and integration testing. Developers wield greater control over ensuring the system's testability, enabling a more thorough examination of individual components and their seamless integration. This increased flexibility empowers developers to

	plugin's business logic may face certain limitations.	implement robust testing practices, ensuring the reliability and stability of the entire system.
Security	Reliant on the security mechanisms established by the Kodi official infrastructure, this method boasts a relative level of security. Nevertheless, developers bear the responsibility of ensuring that the plugin upholds robust security practices when interacting with the API, including vigilant user authentication and secure data transmission. While leveraging the official Kodi security framework, developers must remain diligent in their implementation to fortify the integrity of the plugin.	Developers shoulder the responsibility of handling security concerns independently, encompassing user authentication, data encryption, and safeguarding against potential attacks. The level of security is intricately tied to the proficiency and diligence of the developers in implementing robust security measures. In this method, the onus is on the developers to craft and enforce security protocols, necessitating a meticulous approach to fortify the system against potential vulnerabilities.
Availability	Constrained by the availability of the Kodi official API, the plugin's accessibility is intricately linked to the operational status of the Kodi servers. The method's availability is contingent upon the reliability and functionality of the central Kodi server infrastructure.	The availability of the custom system is subject to the management of the backend servers by the developers themselves. In this scenario, developers bear the responsibility of implementing suitable redundancy and monitoring mechanisms to ensure the system attains and sustains high availability. This necessitates a proactive approach from developers in instituting measures that guarantee uninterrupted service and minimize downtime.
Maintainability	Maintenance of the plugin is relatively straightforward as it relies on the Kodi official API, alleviating the need for frequent backend code upkeep. However, it is imperative to ensure compatibility between the plugin and the API. While the simplicity of maintenance is a notable advantage, developers must remain vigilant in addressing potential compatibility issues to guarantee seamless operation.	The custom system may necessitate more maintenance efforts, including updates to the database structure and adjustments to API endpoints. Nevertheless, developers enjoy greater control and flexibility in adapting to changes. This enhanced control allows developers to proactively respond to evolving requirements, ensuring the system remains adaptable in the face of modifications.

Integration	In Method 1, where Kodi Ratings and Reviews API Integration is employed, the direct utilization of Kodi's official API stands out as a distinctive advantage. This approach ensures a clear and immediate integration pathway, with the use of Python programming language seamlessly aligning with the Kodi architecture. However, it bears the caveat of potential susceptibility to changes in the Kodi API, necessitating vigilant updates to maintain the integrity of the integration. Furthermore, the method imposes user authentication through Kodi accounts, a factor that might introduce an elevated interaction threshold in certain scenarios.	Method 2 involves the creation of a Custom Ratings and Reviews System. This method showcases a higher degree of autonomy by deploying a custom database table, providing increased flexibility and independence from third-party API constraints. The customization of API endpoints in the backend allows for precision in handling frontend requests. While this approach offers greater control, it demands additional backend development effort, potentially introducing complexity into the integration process. The use of popular frontend frameworks like Vue.js or React enhances the user interface's adaptability, yet it requires frontend developers to be well-versed in different frameworks, possibly impacting development costs.

5.5 The Better Approach

When considering non-functional requirements, particularly in the realm of "Integration," the deliberate choice of Approach 2, the Custom Ratings and Reviews System, as the preferred approach for implementing a plugin rating and review system in Kodi is a thoughtfully crafted strategic decision. Firstly, Approach 2, through the creation of a custom database table, endows the plugin with enhanced flexibility, liberating it from the constraints of third-party APIs and allowing for precise adjustments to cater to the plugin's distinct needs. This autonomy provides developers with the capability to fine-tune the system according to the unique characteristics of the plugin. In contrast, Approach 1, relying on the official Kodi API, exhibits a higher sensitivity to API changes, necessitating prompt updates to maintain the coherence of the integration.

Moreover, Approach 2 integrates customizable API endpoints and popular front-end frameworks such as Vue.js or React, presenting users with a visually captivating and intuitively designed user interface. This level of customization not only streamlines communication between the backend and frontend, ensuring efficiency and orderliness but also elevates the overall user experience by creating an engaging interface. In Approach 1, relying on the official API, the degree of customization for the frontend may be comparatively limited, potentially falling short of meeting the unique design requirements of the plugin. Additionally, Approach 2 introduces a more personalized user authentication process, reducing potential interaction barriers compared to Approach 1, where users are required to authenticate through their Kodi accounts. This enhancement in user convenience is a testament to the emphasis on crafting an intuitive and user-friendly environment.

In conclusion, the selection of Approach 2 underscores not only flexibility, customizability, and a reduced reliance on external APIs but also prioritizes the creation of an immersive user experience within the dynamic Kodi ecosystem. This choice accentuates a keen focus on the plugin's specific requirements, reflecting the developer's profound consideration for user interaction, system flexibility, and design aesthetics.

6.0 Effects on Conceptual Architecture

6.1 Effect on high-level conceptual architecture

As it works on the enhancement of the performance of Add-ons, it has a few effects on the relative component. However, there’s no need to consider the player core and Windows manager top-level component.

Approach 1:

Add-ons Rating and Reviews <=> Add-ons: The "Add-ons Rating and Reviews" component in Kodi plays an important role in enhancing the user experience by facilitating a community-driven evaluation system. With this feature, users gain the ability to compare similar add-ons based on community ratings, allowing them to make informed decisions. By gathering the ratings and reviews provided by other users, individuals can assess the stability and reliability of different yet comparable add-ons. This community-driven feedback mechanism not only empowers users to choose the most suitable add-ons for their needs but also fosters a collaborative environment where the collective insights of the Kodi community contribute to the overall quality and utility of the available add-ons. The ratings and reviews become valuable metrics, aiding users in navigating the diverse landscape of add-ons and ensuring a more seamless and satisfying media center experience.

Moreover, it's essential to recognize that the quality of add-ons directly influences their respective ratings and reviews within the "Add-ons Rating and Reviews" component of Kodi. This symbiotic relationship underscores the significance of add-on developers in consistently delivering high-quality and reliable functionalities. Users, in turn, contribute to the ecosystem by assessing and reflecting on their experiences through ratings and reviews. A well-crafted and dependable add-on is likely to garner positive feedback, thereby enhancing its overall rating and reputation in the community. Conversely, any issues related to stability, performance, or functionality may lead to less favorable reviews, impacting the add-on's standing. This dynamic interplay between developers and users fosters an environment of continuous improvement, encouraging developers to refine and optimize their creations, ultimately benefiting the entire Kodi community.

Add-ons Rating and Reviews => GUI elements: The integration of the "Add-ons Rating and Reviews" component with the GUI extends beyond a mere functional connection. This component is intricately designed to seamlessly embed new elements into the graphical user interface, enriching the user experience during the exploration and selection of add-ons. Users can expect to encounter additional graphical elements, including dynamic ratings and detailed review information seamlessly incorporated into the interface. These enhancements not only provide valuable insights briefly but also contribute to a visually cohesive and informative interface, empowering users to make well-informed decisions when navigating through the diverse array of available Kodi add-ons.

Approach 2:

Add-ons Rating and Reviews => Storage: a new node is added to the top-level. Operating equivalently to a database, Storage adeptly manages the influx of user-generated data, creating a structured repository that captures nuanced feedback and evaluations of the diverse user base.

6.2 Effect on low-level conceptual architecture

Game, Picture, Program Add-ons (Approach 1): These low-level architectures must provide access to users to leave their comments and reviews. They literally use some Kodi APIs to provide a rating and reviews community.

Rating database in Storage (Approach 2): Depending on the design, the rating database might store historical rating data, allowing users to view changes in ratings over time. This could include tracking the evolution of an add-on's popularity and performance. The rating database might be utilized for reporting and analytics purposes. Components responsible for generating insights into popular add-ons, user preferences, and trends could leverage the data stored in the rating database.

7.0 Use Cases

7.1 Use Case 1

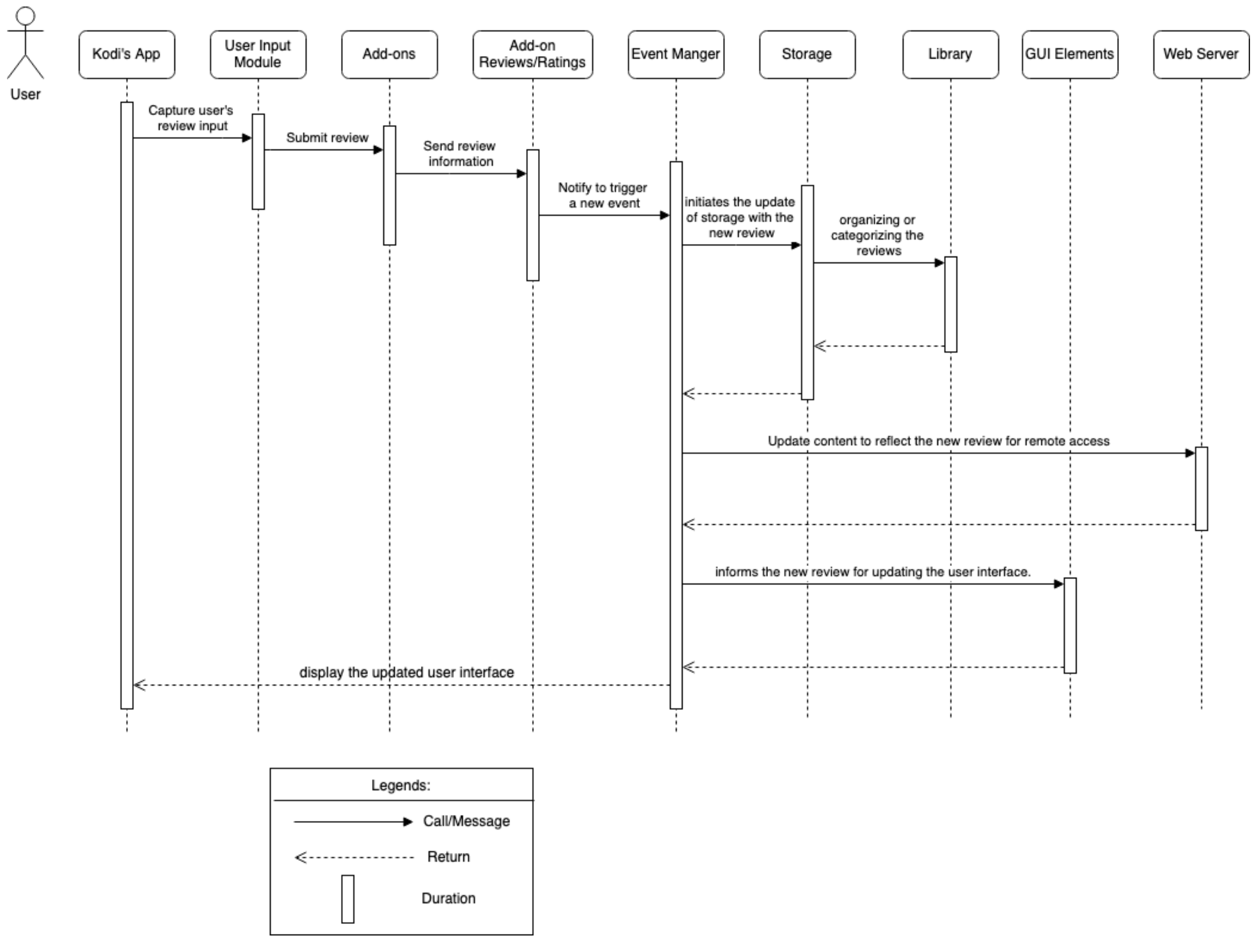
User
Kodi's App
User Input Module
Add-ons
Add-on Reviews/Ratings
Event Manger
Storage
Library
GUI Elements
Web Server
Capture user's review input
Submit review
Send review information
Notify to trigger a new event
initiates the update of storage with the new review
organizing or categorizing the reviews
Update content to reflect the new review for remote access
informs the new review for updating the user interface.
display the updated user interface
Legends:
Call/Message
Return
Duration

Figure 3: Sequence Diagram of "User Submits a Review" under Approach 2

In the use case "User Submits a Review," the process begins with the user interacting with the Kodi application through the User Input Module. Upon selecting an add-on for review, the User Input Module triggers a sequence of events. The Add-on Reviews/Rating Subsystem, responsible for managing user reviews and ratings, receives the user's submission request. The Event Manager plays a crucial role in coordinating the flow of information within the system. It facilitates the storage update by communicating with the Storage component to store the new review. The Library component, responsible for categorizing multimedia content, may also be involved in updating relevant information. Simultaneously, the GUI Element is notified of the user's action, prompting it to update the user interface to reflect the submitted review. The Web Server, responsible for managing remote interactions, may be involved in syncing the review data to ensure consistency across the platform. The entire process is orchestrated seamlessly, allowing users to contribute reviews effortlessly, enhancing the overall user engagement and community-driven aspect of the Kodi platform.

7.2 Use Case 2

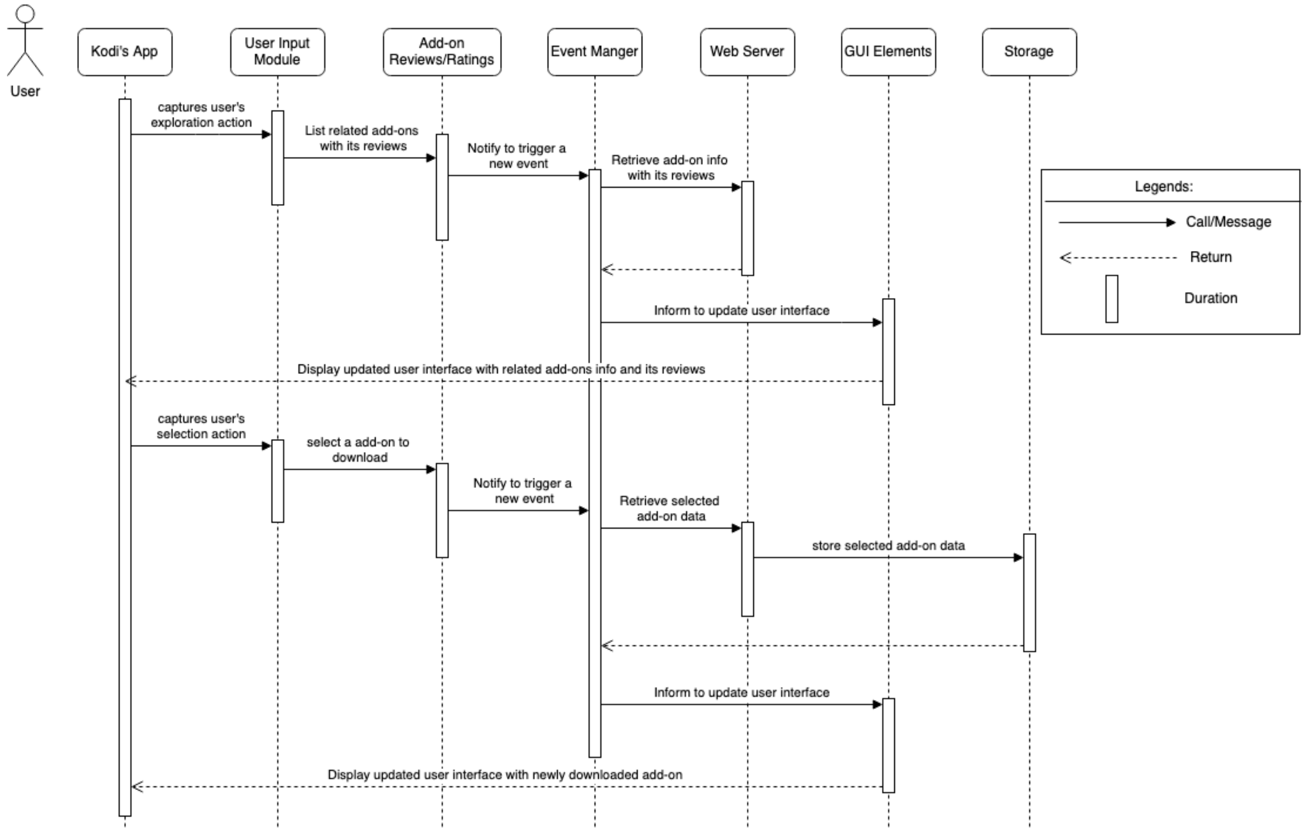
User
Kodi's App
User Input Module
Add-on Reviews/Ratings
Event Manger
Web Server
GUI Elements
Storage
captures user's exploration action
List related add-ons with its reviews
Notify to trigger a new event
Retrieve add-on info with its reviews
Legends:
Call/Message
Return
Duration
Inform to update user interface
Display updated user interface with related add-ons info and its reviews
captures user's selection action
select a add-on to download
Notify to trigger a new event
Retrieve selected add-on data
store selected add-on data
Inform to update user interface
Display updated user interface with newly downloaded add-on

Figure 4: Sequence Diagram of "User Explores and Selects an Add-on to install" under Approach 2

In the user case of "User Explores and Selects an Add-on to install" within the Kodi application, the User Input Module serves as the initial point of interaction. When the user explores or searches for add-ons, the User Input Module captures these actions. Subsequently, a request is sent to the Add-on Reviews/Ratings Subsystem to list related add-ons along with their reviews. The Add-on Reviews/Ratings Subsystem, upon receiving the request, notifies the Event Manager to trigger a new event for retrieving add-on information with reviews from the Web Server. Once the data is fetched, the Event Manager informs the GUI Element to update the user interface. The GUI Element, now armed with the latest information, displays an updated interface featuring details about the add-ons and their reviews. The user, equipped with this valuable insight, can then select a preferred add-on for download. The User Input Module captures this selection action, leading to a request for downloading the chosen add-on. The Event Manager, in response, triggers another event to retrieve specific data for the selected add-on from the Web Server. The acquired add-on data is stored in the Storage component, and the Event Manager communicates with the GUI Element to update the user interface. The final result is a seamlessly updated user interface, displaying the newly downloaded add-on based on the user's selection, contributing to an intuitive and efficient user experience within the Kodi ecosystem.

8.0 Potential Risks and Limitations

Although adding an addon rating and review system in Kodi brings many benefits, some aspects need to be paid attention to.

First, insufficient user engagement may cause the system to fail to perform as expected. If only a few users are willing to rate and review, the information provided may not be enough to help other users make informed decisions. Additionally, ratings and reviews may be affected by subjective bias or be dominated by a small number of active users, creating a misleading impression.

Secondly, in terms of security, processing user ratings and comments means processing more personal information, which requires strict data protection measures, such as encrypted storage and secure data transmission. In addition, special attention needs to be paid to the risk of malicious behavior, for example, users may post spam or defamatory content through the rating system.

Third, in terms of maintainability, with the increase in user-generated content, Kodi needs to effectively manage and review these contents, and regularly update and maintain the comment system.

Finally, in terms of performance, processing and storing large amounts of data may increase the burden on the server and affect response speed. Therefore, database queries are optimized to maintain system performance.

9.0 Lessons Learned

This report taught us how important it is to incorporate user feedback channels into software ecosystems. The addition of the "Add-on Reviews and Ratings" function to Kodi has improved community engagement and information sharing in addition to improving user experience. The process of bringing this innovation to market highlighted the need for user-centered design and demonstrated how crucial it is for software developers to handle technical issues like server load and database consistency. Furthermore, the paper demonstrated how thorough research and meticulous planning might be used to accomplish a balanced integration of new features with the current system design. These observations are important for comprehending how user feedback affects software development and community building, as well as considering different aspects of the technical implementation process.

10.0 Conclusion

The "Add-on Reviews and Ratings" feature is proposed in this study for the Kodi media player application. Based on user feedback, it is intended to provide customers with wiser decision-making support while selecting and installing add-ons. The paper examines the implementation process and its potential multidimensional impacts, as well as the risks and problems that may be encountered, such as database consistency, server load, and the construction of a content management system. This feature not only solves a functional void in Kodi, but it also contributes to a more collaborative and user-friendly experience.

11.0 Data Dictionary & Naming Conventions

Rating database: Data Storage for User Ratings

12.0 References

Kodi Community. "Archive:PAPlayer." Kodi Wiki, 20 July 2020, Archive:PAPlayer. Accessed <https://kodi.wiki/view/Archive:PAPlayer>.

Kodi Wiki. Official Kodi Wiki. (2023). https://kodi.wiki/view/Main_Page
News. Kodi. (n.d.). https://kodi.tv/blog/
Paul, R. (2009, December 29). *XBMC 9.11 makes your open source home theater look shinier*.
Development. Official Kodi Wiki. (n.d.) https://kodi.wiki/view/Development
Dreef, K., Reek, M. van der, Schaper, K., & Steinfort, M. (2015, April 23). *Architecting software to keep the lazy ones on the couch*. Kodi. https://delftswa.github.io/chapters/kodi/