Pesticide application management toolset for improved worker protection

Chathurika Narayana [1], Nathan Thorson [2], Joe D Luck [1]

[1] University of Nebraska-Lincoln, Lincoln, Nebraska, USA

[2] University of Nebraska–Lincoln, Eastern NE Research & Extension Center, USA

A paper from the Proceedings of the
16th International Conference on Precision Agriculture
21-24 July 2024
Manhattan, Kansas, United States

Abstract.

The practice of pesticide use has been widely adopted by production agriculture to maximize yields since the 1950s. Even though it provides beneficial economic returns to the farmers, it also enhances the risk of environmental pollution and is directly associated with the risk of poisoning to agricultural workers. So, the risks associated with worker exposure to pesticides are a significant concern, considering the probability of an individual experiencing long-term health effects and or fatality from a single exposure event. Therefore, the pesticide application management toolset, along with the worker protection standards in the agricultural platform, are necessary.

While adhering to United States Federal Environmental Protection Agency (EPA) Worker Protection Standard (WPS) guidelines, the current systems need considerable time to provide crucial information. These systems rely heavily on human communication as pesticide applicators need to document application details on tabular forms just before application, and access is delayed with centralized databases until the end of the workday. This time lag leaves farm workers vulnerable and reliant on a communication chain for safety information, introducing potential human errors.

The "spray-safely application" development methodology involves collecting fieldwork production data using wireless data transfer technologies. This data will be transferred to the machine data acquisition service (MDAS), a cloud-based storage service provided by agricultural equipment manufacturers. The MDAS contributes with users to document, visually display, and notify field workers who may be at risk for pesticide exposure by integrating a third-party database system that provides safety information related to individual pesticides.

Application testing results imply that the data acquisition aspect quickly determined that the system was successful and could promptly process and return this information to the front end.

The front end also proved to be a success in that, once the data was acquired, it successfully retrieved this information and loaded the necessary webpage. Furthermore, the application can present pesticide information with high-resolution spatial accuracy and provides a popup window, giving users a detailed view of product information.

In conclusion, the proposed "spray-safely application" showcases the potential of precision agriculture technologies to enhance safety and efficiency. It enhances accessibility, reduces delays, minimizes the reliance on human-mediated communication, and provides the information and notification processes necessary for worker protection in a timelier format, which systems currently experience between pesticide application and reporting. With further testing and development, the tool will align with the goals of precision agriculture and set the stage for future advancements that create safer working environments for our agricultural workforce.

Keywords.
Precision agriculture, pesticide documentation, information accessibility, worker protection, agricultural technology.

1 Introduction

The rise of precision agriculture has reshaped farming methods, facilitating the optimized management of agricultural resources through the seamless integration of advanced technologies. However, the documentation and reporting of pesticide applications have not fully leveraged these technological advancements, posing significant risks to agricultural workers. Current pesticide application documentation systems operate on outdated procedures involving manual recording of application details on tabular forms and delayed reporting to centralized databases. This process creates an information accessibility gap, extending up to 24 hours, during which time workers may be exposed to harmful pesticides without adequate information on recent applications.

The need for a more efficient system is underscored by the limitations of current practices. Pesticide applicators often document application details just prior to spraying, and this information is not readily accessible to farm workers until the end of the workday. This delay is compliant with the U.S. Environmental Protection Agency's (EPA) Worker Protection Standard (WPS), which mandates that pesticide application information must be documented and available at a central location within 24 hours post-application (Fults,2016). However, the reliance on manual communication chains increases the risk of human error and delays, potentially exposing workers to pesticide-related hazards.

It was shown that between 2006-2010 there was an annual national average of over 20,000 pesticide exposure cases that required health care treatment. In Nebraska alone - between 2000 and 2009 - the average number of pesticide worker exposure cases was 54 per year. The number of cases in 2009 decreased by roughly half of those in 2000; however, this number was still 2-3 times higher than the national average (Stover, 2014). In 2011, Nebraska saw a decrease in pesticide poisoning cases, with only 34 cases that year, yet this number was still higher than the national average (Nebraska Occupational & Health Surveillance Program, 2014).

The evolution of pesticide application documentation systems has significantly influenced the development of tools to improve worker protection standards in agriculture. Historical efforts, such as the United States Geological Survey (USGS) department's attempt in Washington state in 1994, laid foundational concepts for integrating geospatial information systems (GIS) into pesticide documentation. The USGS system, though advanced for its time, faced several limitations, including system complexity, lack of field resolution, and limited reporting functions, making it impractical for widespread adoption among agricultural producers (Schurr, 1994). This system's inability to document pesticide application with finer resolution than a Public Land Survey System (PLSS) grid and its restricted reporting capabilities illustrate the challenges faced in early precision agriculture initiatives.

More recent developments have focused on addressing specific issues, such as spray drift and natural resource buffer restrictions. For instance, a toolset developed in 2017 aimed to predict areas susceptible to spray drift by incorporating variables like wind direction, wind speed, temperature inversions, equipment boom height, and spray droplet size (Atay & Ayebare, 2017). While this tool effectively utilized geospatial data to visualize potential spray drift areas, it lacked a critical component: the evaluation of its predictive accuracy through field validation. The absence of a thorough assessment highlights a gap in ensuring the reliability of such tools for practical use in mitigating pesticide exposure risks.

The existing literature on pesticide application documentation systems reveals several critical insights and gaps. Early systems demonstrated the potential of GIS in pesticide documentation but were hampered by technological limitations and user adoption barriers. More recent tools have advanced the application of geospatial data but often lack comprehensive validation and time-sensitive reporting capabilities. None of these systems have effectively utilized the wireless transfer of machine data to cloud-based storage for real-time documentation and analysis. This

could significantly enhance compliance with EPA Worker Protection Standards and improve communication among stakeholders in precision agriculture.

Precision agriculture, with its high-resolution data collection and real-time data transfer capabilities, presents an opportunity to enhance pesticide application documentation systems. The industry has seen substantial adoption of precision agriculture technologies, with over 60% of agricultural producers in Kansas using such systems by 2017 and 80% of producers in Nebraska managing their data (Miller,Griffin, Ciampitti, & Sharda, 2019)(Cornhusker Economics Agricultural Economics, 2015). These advancements indicate a readiness within the industry to adopt more sophisticated data management practices, including real-time data transfer, which could significantly reduce the risks associated with pesticide use.

The primary goal of this research is to develop a Pesticide Application Management Toolset that aligns with WPS requirements while leveraging real-time data transfer technologies to minimize the risk of pesticide exposure to agricultural workers. This toolset aims to improve the timeliness and accuracy of pesticide application documentation, thereby enhancing worker safety. By integrating GIS formats, wireless data transfer, and centralized databases, the proposed system seeks to provide immediate access to application specifics, ensuring compliance with regulatory standards and reducing reliance on error-prone manual processes.

2 Materials and methods

2.1 Overview

The methodology for the "Spray-Safely" Pesticide Application Management Toolset for Worker Protection Standards focuses on integrating advanced precision agriculture technologies to enhance the safety and protection of agricultural workers. Precision agriculture technologies have seen rapid advancements, with only 12% of surveyed Nebraska producers not collecting information on their production facilities (Cornhusker Economics Agricultural Economics, 2015). The system leverages wireless data transfer technologies via 4G and upcoming 5G networks, facilitated by Modular Telematics Gateways that connect to agricultural equipment providers' cloud-based data storage services. These services have made significant strides towards cross-brand compatibility through data consortiums and software development initiatives such as the ("JohnDeereADAPTPlugins," 2021). Utilizing a Machine Data Acquisition Service (MDAS) API alongside a third-party product database API, the system documents visualizes and notifies field workers about real-time pesticide applications. The system incorporates a database to track field worker information, including EPA Worker Protection Standard (WPS) training records, and employs security access permissions to manage user access. The Spray-Safely application, accessible via an internet browser, integrates data from various sources to reduce reporting latency and improve communication among agricultural workers. The system's workflow, depicted in Figure 1, illustrates the process from field data collection to user notification, ensuring compliance with regulatory standards and promoting a safer working environment.

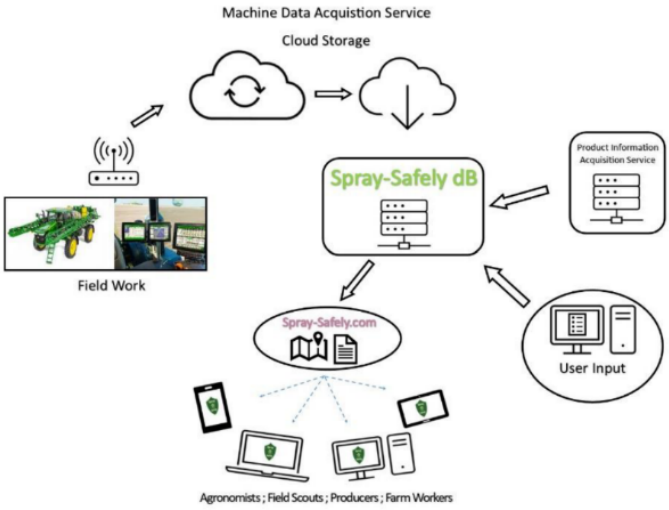


Figure 2.1.1. Spray Safely System Overview (Thorson, N. W,2022).

2.2 Application Frontend Design and Development

The design and development of the frontend for the "Spray-Safely" involved a strategic approach to ensure user accessibility, compliance with regulatory standards, and seamless integration with backend functionalities. The frontend was designed prior to the backend development to accurately identify the necessary backend deliverables, thereby ensuring the system could meet all user needs effectively.

Design Phase

The initial step in the front-end design focused on identifying key information required by users, primarily driven by the EPA's WPS requirements. This included details such as the name and active ingredients of the pesticide products applied, EPA registration numbers, return entry intervals (REI), crop or site treated, location and description of treated areas, and the dates and times of pesticide applications (Fults, 2016). The design needed to cater to various users, including producers, applicators, and crop scouts, emphasizing automation to reduce reliance on manual data entry. Due to the convenience factor, significant effort was directed toward automating as many data acquisition features as possible. The system would also need to handle instances allowing the user to input application information or update previously acquired information manually. The platform selection was also critical in ensuring broad accessibility and feature development. Despite the benefits of a mobile application for offline use and device-specific functionality, the decision was made to develop a browser-based application. The front end was developed using HTML5, CSS3, and JavaScript, leveraging Bootstrap for managing interoperability and ensuring responsive design across different screen sizes.

Incorporating geospatial features was essential to the application's functionality, allowing users to visualize and manage pesticide application data spatially. The Leaflet library was chosen for its extensive documentation and feature set, enabling users to create and display field boundaries and other relevant data. ESRI's JavaScript library provided the base map services, offering the most recent and frequently updated aerial imagery (ESRI).

Development Tools and Process

Development tools such as Bootstrap Studio facilitated the initial design, providing an integrated development environment (IDE) for managing HTML, CSS, and JavaScript. However, for more efficient coding, Notepad++ was later used for direct code entry. Node.js and Python Flask were employed for local network hosting and testing, ensuring functionality and responsiveness of the web application.

The farm manager tree displayed the farms, fields, and applications that were associated with the currently logged-in user. From there, the user could select an application and navigate to the field without physically panning across the map to the desired field for inspection. Additional options contained in the farm management tree were four buttons that allowed the user to input application information. These four buttons were comprised of adding a farm, a field, a boundary, and a pesticide application, respectively. All buttons open modal windows, displayed over top of the current webpage, containing the necessary user input features to capture the desired information. modals are essentially windows or dialog boxes that pop up and display over the current webpage, thus deactivating all webpage content and functionality to draw the user's focus (Juviler, 2021).

The front end was designed to interact seamlessly with the backend, utilizing POST and GET methods over the HTTP protocol to transmit and receive data. Upon loading, the frontend acquired geojson data from the backend, which was then used to populate the management tree and other geospatial features. The use of the Leaflet library allowed for detailed mapping and interaction with pesticide application data, ensuring that users could easily navigate and access relevant information.

2.2 Application Backend Design and Development

The design and development of the frontend for the "Spray-Safely" involved a strategic approach to ensure user accessibility, compliance with regulatory standards, and seamless integration with backend functionalities. The frontend was designed prior to the backend development to accurately identify the necessary backend deliverables, thereby ensuring the system could meet all user needs effectively.

Design Phase

The backend for the "Spray-Safely" Toolset was designed with a strong focus on data acquisition and storage, which necessitated a comprehensive design process for the database structure. This included defining attributes, attribute data types, tables, table structures, and the necessary relationships between these entities. The database needed to store information on pesticide applications, user accounts, and pesticide products. Given the complexity and volume of the data, a feature-rich database program was essential, leading to the selection of MySQL for its robustness and functionality.

The MDAS API played a crucial role in the data acquisition process, providing extensive documentation and sample code snippets. While not all API procedural calls were necessary for our project, a significant subset was required. These included calls for initial authorization, service provider account organization information, farm information attributed to the account, and field information attributed to each farm. Additionally, the ability to handle multiple organizations within the service provider's system was critical, necessitating a database structure capable of storing and interconnecting this information.

The API service provider also offered a data subscription service, akin to a webhook, which would notify our backend of changes within the user’s account. This feature required additional data storage to ensure proper operation with the MDAS. Key data elements to be stored included access levels, access tokens, refresh tokens, token expirations, organization identification numbers, organization names, organization types, farm names, and identification numbers, field names and identification numbers, subscription tokens, subscription client keys, subscription

identification numbers, subscription names, and subscription filters. For application data, the system needed to store application dataset identification numbers.

Primary keys were essential for establishing relationships between database entries, ensuring the system could accurately interconnect data, such as identifying which fields belong to a specific farm, which farm to a specific organization, and which organization to a specific user account. This hierarchical structure also supported determining which user account a specific subscription service call was regarding and where the application dataset should be stored.

The database design included several key tables to manage the extensive data requirements. The *User_Info* table stores user account information, with fields indicating edit access, account confirmation status, and organization ownership. The *Org_Info* table manages organization data within the Spray-Safely application, while the *JD_Org_Info* table handles organization and event subscription data from the MDAS API. The *Farm_Info* table stores farm information, both manually created and imported from the MDAS API. Similarly, the *Field_Info* table stores field information, including GeoJSON data for field polygons. The *App_Info* table manages pesticide application data, both manually created and imported from the MDAS API. The *Applicator_Info* table stores information on applicators, including licensing details. The *Tank_Mix_Info* table manages tank mix information defined by users and the *Carrier_Info* table stores carrier information for pesticide applications. Lastly, the *Product_Info* table is populated via a CRON job that retrieves data from a Product Information Acquisition Service (PIAS)database, including REI times and descriptions. This comprehensive table structure ensures efficient data management and supports the complex relationships necessary for the toolset's functionality.

Given the extensive data storage requirements and the complexity of relationships, Python Flask was chosen as the main framework language for backend development. Flask's compatibility with MySQL and the MDAS API made it an ideal choice for carrying out the necessary database operations and ensuring seamless integration with external data sources.

For a detailed representation of the database structure, refer to the “Spray-Safely DB ER Diagram” in the Appendix. This ER diagram outlines notable tables, their relevant columns, and their interrelationships, ensuring a comprehensive understanding of the backend design necessary for managing pesticide application information in compliance with Worker Protection Standards.

Development Tools and Process

To develop the backend for this tool, it was essential to establish a reliable development environment. The first step involved installing and upgrading the version of Python to the stable version, Python 3.8. Once these updates were completed, the necessary Python libraries were installed. With the development environment properly configured, the backend development could proceed. A pre-developed file, supplied by the MDAS with their sample code, was used to start the Flask service, ensuring smooth integration with the API. Detailed instructions and the required library packages for the backend Python Flask environment are provided on the MDAS GitHub sample code page ("MyJohnDeereAPI-OAuth2-Python-Example," 2020).

The connection process between the Spray-Safely application and the MDAS service begins with the authorization process. The procedure initiates an authorization request to the MDAS service, redirecting the user to the MDAS website to grant the Spray-Safely application access to their MDAS organization. Upon user approval, the MDAS service makes a callback to the Spray-Safely backend, requiring the development of a callback procedure. The callback procedure starts by retrieving the username from the browser session to identify the user. It then uses the returned OAuth access code to request access and refresh tokens from the MDAS service. These tokens are stored in the database along with an expiration time, as the MDAS changes the access token every 12 hours. Before any data request to the MDAS is made, the token's validity is checked and refreshed if necessary.

Data acquisition from the MDAS API follows a structured process. The procedure begins by acquiring the organization, farm, and field data associated with the user, leveraging the MDAS pagination process to handle large datasets. Sub-procedures form the request parameters in JSON format, which are then used to make API calls. After obtaining the necessary data, a subscription service is created to reduce API traffic and ensure timely data updates. The `receiveEvents` procedure handles incoming event notifications from MDAS, using multi-threading to start a new process for data acquisition and storage while immediately returning a 204 No Content response.

To manage geospatial data, a boundary generator process converts geospatial point data into a polygon encompassing all data points. This involves using the concave hull generating library, which requires adjustments for compatibility with the Python environment. Detailed steps for implementing this library are outlined in Appendix B. The boundary-generating process begins by reading the shapefile dataset and parsing the geometric features. These features are then passed to the concave hull library to generate a polygon, which is buffered and returned to the backend. This process is crucial for accurately representing application areas within the Spray-Safely database.

The subscription service for application data ensures that the backend is notified of relevant events, such as adding or removing fields or uploading application data. The `receiveEvents` procedure uses multi-threading to handle data acquisition and storage asynchronously, ensuring prompt responses to MDAS while processing data in parallel. This setup significantly reduces the need for repetitive API calls and ensures that the application data is updated in real time.

For frontend requests, the backend queries the database for pesticide application entries, formats them to meet JSON standards, and returns them in JSON format. Since the MDAS returns operational information in a specific format, database entries must be reformatted before being sent to the front end. The product information is processed through the *eval* function to ensure compatibility with JSON standards, concatenated with additional data, and returned using the *jsonify* function. This approach prevents parsing errors and ensures data integrity.

3 Results

The testing of the backend data acquisition process for the “Spray Safely” utilized two methods. The first method involved physically uploading datasets to the MDAS organization, triggering the backend data acquisition process. While this method was representative of a machine data transfer, requiring many datasets to thoroughly troubleshoot and confirm backend operations, it was complemented by a second method. This second method involved reprocessing datasets that had already been uploaded to the MDAS system, also triggering a call to the Spray-Safely backend acquisition process. This combination reduced the number of datasets needed for adequate testing.

For this portion of the project, twelve pesticide application files were utilized, with raw dataset uploads ranging from 35 KB to 430 KB (covering areas from 8 acres to 163 acres). The datasets included fields that were both partially and entirely applied. The testing demonstrated that the system was successful in data acquisition, processing, and timely return of information to the front end. On average, the MDAS service triggered a call to the Spray-Safely backend within 1-3 seconds of dataset upload. This method closely simulates a wireless data transfer process, verifying the system's capability to process datasets and trigger data acquisition events within 1-3 seconds of receiving the dataset from the machine.

Reprocessing datasets on the MDAS system exhibited greater variability in event trigger timing, ranging from 1-3 seconds to 1-2 hours after initiating the reprocessing procedure. This variability

is believed to be due to the MDAS service prioritizing dataset uploads over reprocessing. During off-peak times, such as Sundays and late evenings, more timely responses were observed, likely due to lower machine data traffic.

Upon receiving the event trigger, the Spray-Safely backend processed the datasets, making application entries available for frontend retrieval within an average of 6-7 minutes. This time varied based on dataset size, with larger datasets requiring more time. For instance, the longest processing time for a single entry was 24 minutes for a 38-acre dataset, while other tests averaged 14.5 minutes with a standard deviation of 1 minute 38 seconds for similar-sized datasets.

One limitation encountered during testing was related to geometric boundary calculations, primarily due to the limited random-access memory (512 MB) of the hosting server. This issue was significant when processing larger datasets, such as a 426 KB dataset representing a 160-acre field. This dataset contained 345,461 geospatial points, which overwhelmed the server's memory during the concave hull exterior boundary generation process. Future improvements could involve increasing server memory and adjusting dataset acquisition resolution to alleviate this issue.

Another limitation arose from the concave hull library process, which only creates polygons with external boundaries. This caused failures with datasets representing incomplete field operations, common in agricultural pesticide applications when operators need to refill tanks or make changes to their monitors mid-application. The presence of large gaps in data points resulted in boundary generation failures. Addressing this limitation would require enhancements to the boundary generation procedure to handle incomplete datasets more effectively.

The front end successfully retrieved, parsed, and displayed the acquired data, populating the farm management tree with panning capabilities (Figure 3.1) (Thorson, N. W,2022). The pesticide application information was displayed with high spatial resolution and accessible via popup windows and models for increased product information (Figures 3.2 & 3.3) (Thorson, N. W,2022).

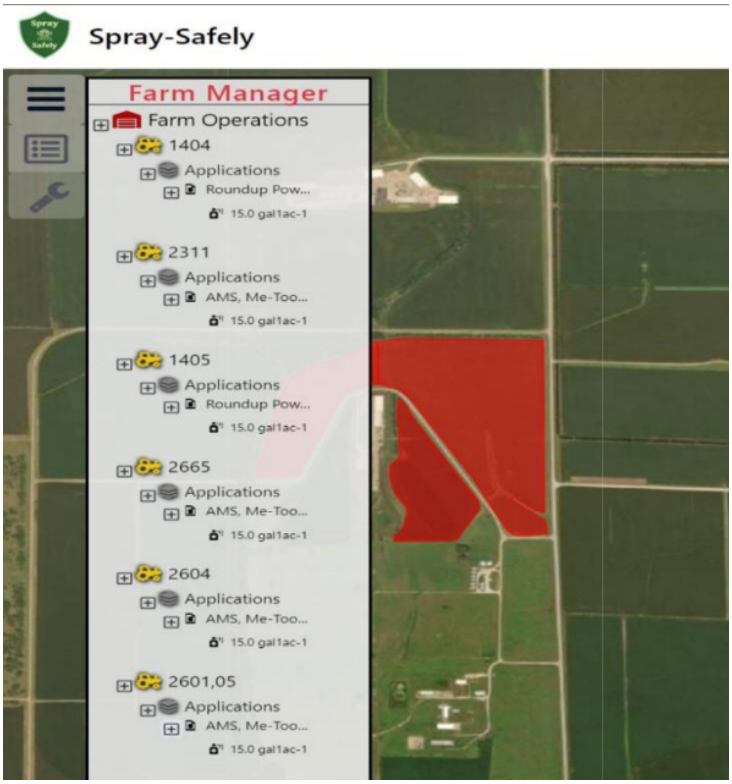


Fig 3.1 Spray Safely Management Tree (Thorson, N. W,2022)

Fig 3.2 Spray Safely Feature Popup (Thorson, N. W,2022)

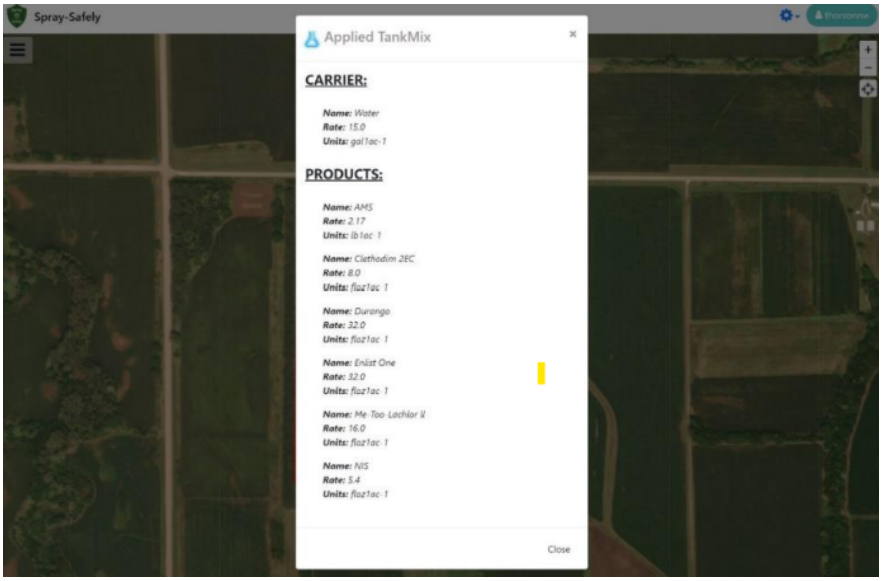


Fig 3.3 Spary Safely Product List Modal (Thorson, N. W,2022)

Despite some limitations, the project succeeded in accurately representing pesticide applications at a finer resolution than previous documentation attempts, such as the USGS database application system from 1994, which documented pesticide applications on a 40-acre section

basis. The Spray-Safely application refined this process to sub-foot accuracy, enhancing user convenience and data acquisition timeliness through automated input and wireless data transfer.

4 Conclusion or Summary

The development and testing of the Spray-Safely application have demonstrated significant advancements in accurately portraying the spatial location and specifics of pesticide applications in a timely and efficient manner. Utilizing current technological advancements, the project successfully automated several processes, including information capture and spatial representation, enhancing convenience for producers, machine operators, and end users. By leveraging these advancements, the Spray-Safely application facilitates seamless information relay from the field to the field workers without the need for direct communication, thus reducing human error susceptibility.

Despite the project's successes, the limited testing identified several areas for improvement and additional functionality that could enhance the utility of the Spray-Safely tool. These insights underscore the potential benefits of continued development to address current limitations and integrate new features, ultimately aiming to create a comprehensive stand-alone program for documenting pesticide applications. Future work will focus on further refining the toolset, ensuring its adherence to all aspects of worker protection standards.

To enhance the Spray-Safely toolset, future development should focus on resolving issues related to data acquisition procedures and the boundary-generating toolset. One significant improvement would be utilizing the `multiPolygon` function from the Shapely library instead of the current `polygon` function. This function can handle datasets with missing point data, preventing procedural failures during boundary processing. An alternative approach could involve segregating point clouds from the dataset, running the boundary-generating tool on each cloud, and merging the results into a single polygon feature representative of the operation area.

Addressing the dataset size limitation is another crucial area for future work. Transitioning from a per-section to a per-point dataset acquisition method could significantly reduce dataset size and, consequently, the required random-access memory. For instance, reducing the data size from 345,461 points to 21,591 points for a 163-acre field would mitigate memory constraints. Additionally, moving to a virtual private server with more resources or upgrading the current hosting platform could alleviate these limitations.

Further development should also focus on enhancing backend procedures to connect data input windows on the front end to the Spray-Safely databases. Utilizing HTTP protocols to pass user input to backend Flask procedures, which then interface with the database management Python libraries, would streamline this process.

Another area for future work involves developing procedures to acquire pesticide product information from the PIAS, which was instrumental in designing the Spray-Safely procedures and databases. This enhancement would increase the availability of crucial information to end users, reducing the risk of pesticide exposure. This effort would entail creating procedures for acquiring, storing, and managing product information in the Spray-Safely databases.

Finally, beta testing the software with a stakeholder group, including producers, farm workers, and crop scouts, is crucial for assessing the software's ease of use, convenience, reliability, and added value. Collaborating with a university extension outreach program, this testing will gather stakeholder feedback and identify unforeseen issues, guiding further development and feature enhancement.

The Spray-Safely application project has made significant strides in automating the documentation and spatial representation of pesticide applications, providing a reliable and user-friendly tool for producers and field workers. While the project has identified several limitations,

the outlined future work provides a clear path for addressing these issues and enhancing the tool's functionality. With continued development and testing, the Spray-Safely toolset can become an essential component in adhering to worker protection standards and reducing pesticide exposure risks in agricultural settings.

Acknowledgments

I extend my deepest gratitude to N. W. Thorson, NFARMS Research Manager at the University of Nebraska–Lincoln, whose foundational work, "Pesticide Application Management Toolset for Worker Protection Standards" (2022), provided the essential framework for this study. As a co-author, N. W. Thorson's expertise and dedication significantly enriched this work.

References

Atay, C. E., & Ayebare, P. (2017). Determination of Buffer-Zones using Agricultural Information System. TEM Journal, 6(2), 363-371. doi:10.18421/TEM62-23

Cornhusker Economics Agricultural Economics, D. (2015). DigitalCommons@University of Nebraska - Lincoln. In

Fults, J. (2016). EPA WPS - How to comply with 2015 revised wps for ag pesticides.

JohnDeereADAPTPlugins. (2024). Retrieved from https://developer.deere.com/offline-sdk

Juviler, J. (2021). Retrieved from https://blog.hubspot.com/website/modal-web design#:~:text=A%20modal%20(also%20called%20a,action%20or%20by%20closing%20it.

Miller, N. J., Griffin, T. W., Ciampitti, I. A., & Sharda, A. (2019). Farm adoption of embodied knowledge and information intensive precision agriculture technology bundles. Precision Agriculture, 20(2), 348-361. doi:10.1007/s11119-018-96114

MyJohnDeereAPI-OAuth2-Python-Example. (2020). Retrieved from https://github.com/JohnDeere/MyJohnDeereAPI-OAuth2-Python-Example.

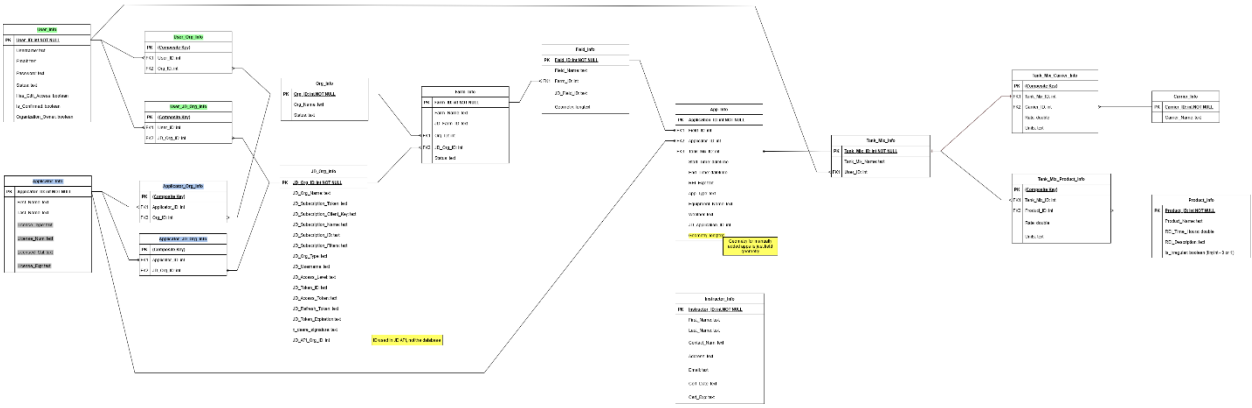
Nebraska. Occupational, S., & Health Surveillance Program, a. i. b. (2014). Nebraska occupational health indicator report, 2014 / Occupational Safety and Health Surveillance Program, Nebraska Department of Health and Human Services. In (pp. 0-0). Lincoln, NE: Nebraska Department of Health and Human Services.

Thorson, N. W. (2022). Pesticide Application Management Toolset for Worker Protection Standards. https://core.ac.uk/download/549024176.pdf

Schurr, K. (1994). Documentation of a spatial data-base management system for monitoring pesticide application in Washington. Tacoma, Wash.: Denver, Colo.: U.S. Dept. of the Interior, U.S. Geological Survey; USGS Earth Science Information Center, Open-File Reports Section [distributor].

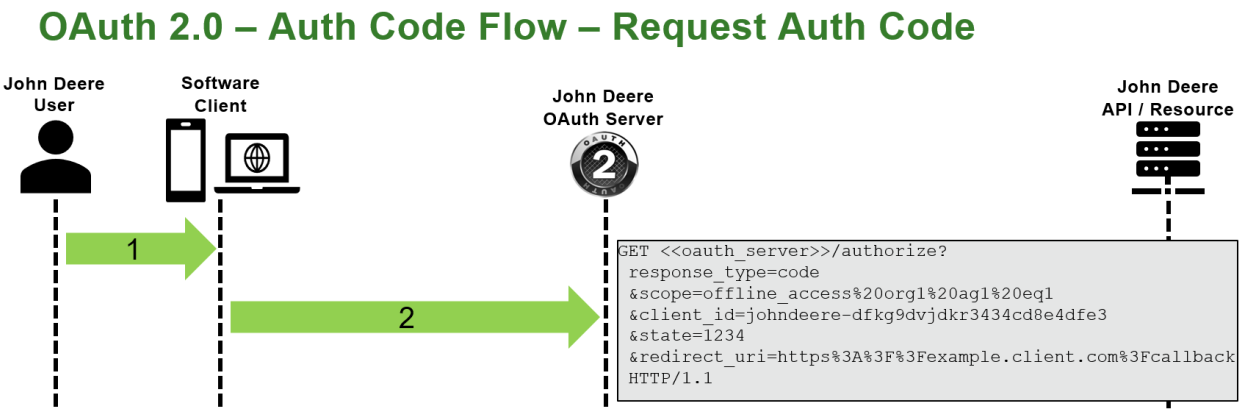
Stover, D. (2014). Nebraska occupational health indicators, 2000-2009 / prepared by Derry Stover, Nebraska Department of Health and Human Services, Division of Public Health, Office of Epidemiology, Occupational Safety and Health Surveillance Program. In (pp. 0-0). Lincoln, NE: Nebraska Department of Health and Human Services.

Appendix A: Spray-Safely DB ER Diagram



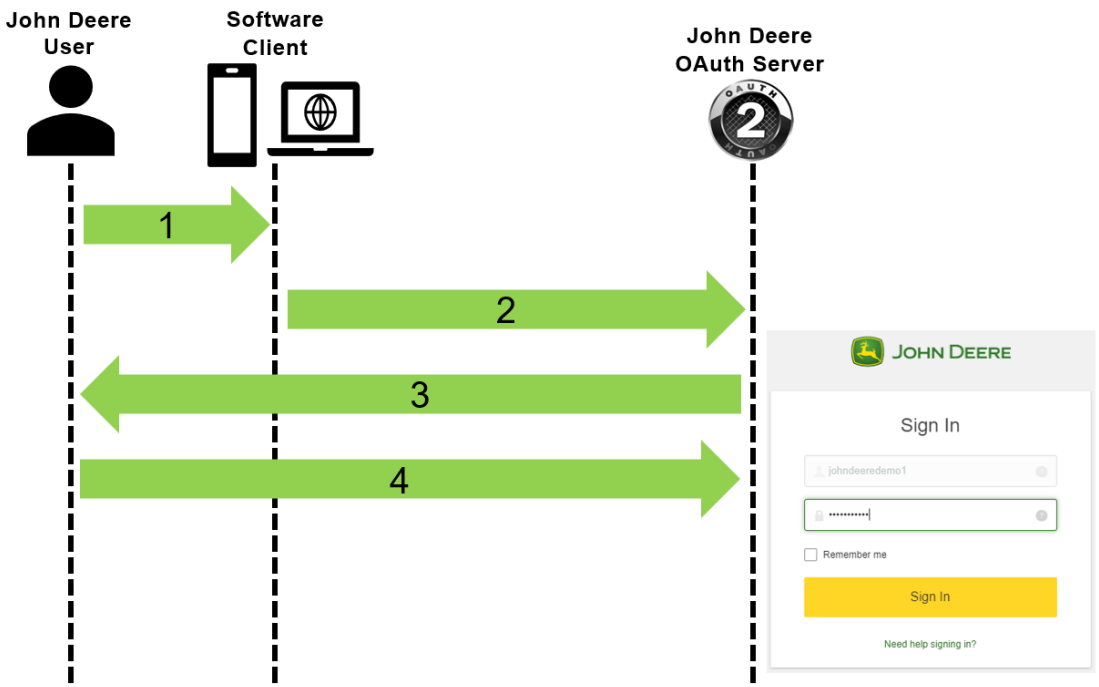
Appendix B: ("Develop With Deere," 2022)

Steps 1 & 2 - The customer initiates a request for data from a client application, and the client sends OAuth request to the authorization server with the proper headers.



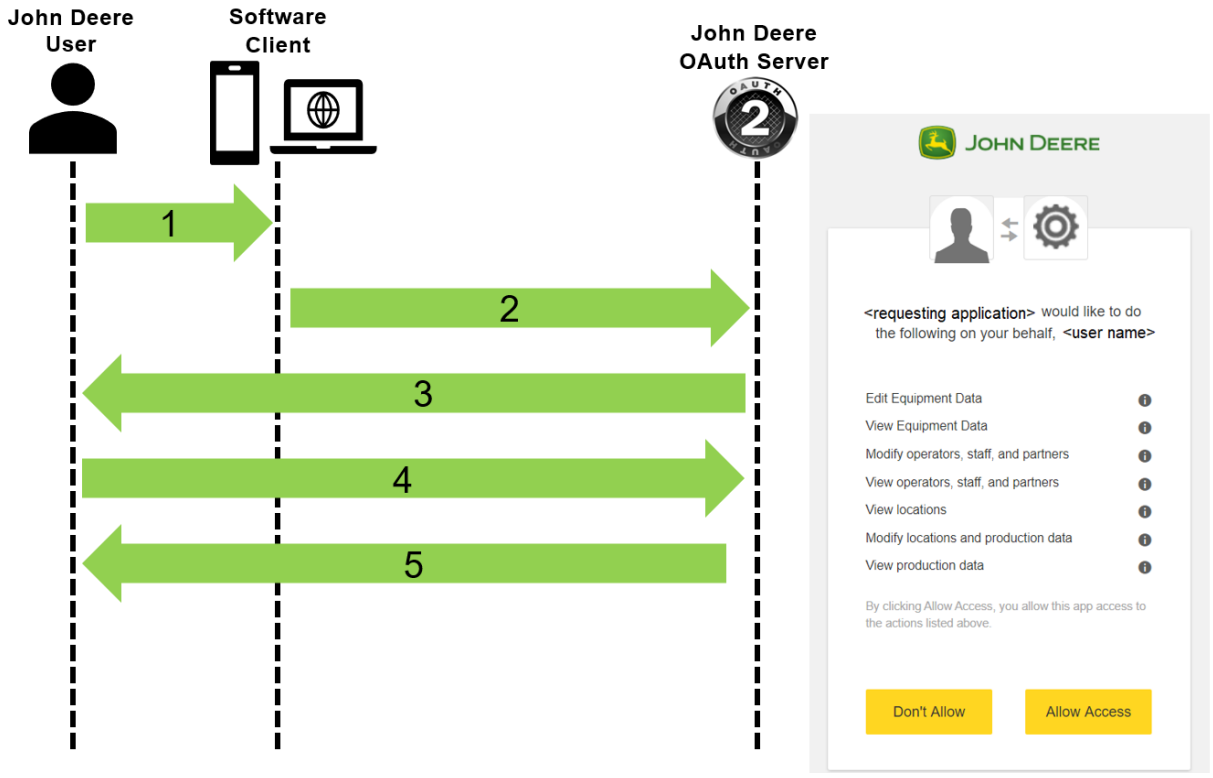
Step 3, 4 - The customer is redirected to John Deere sign-in page. Then signs into John Deere, and the request is redirected back to the authorization server.

OAuth 2.0 – Auth Code Flow – Request Auth Code

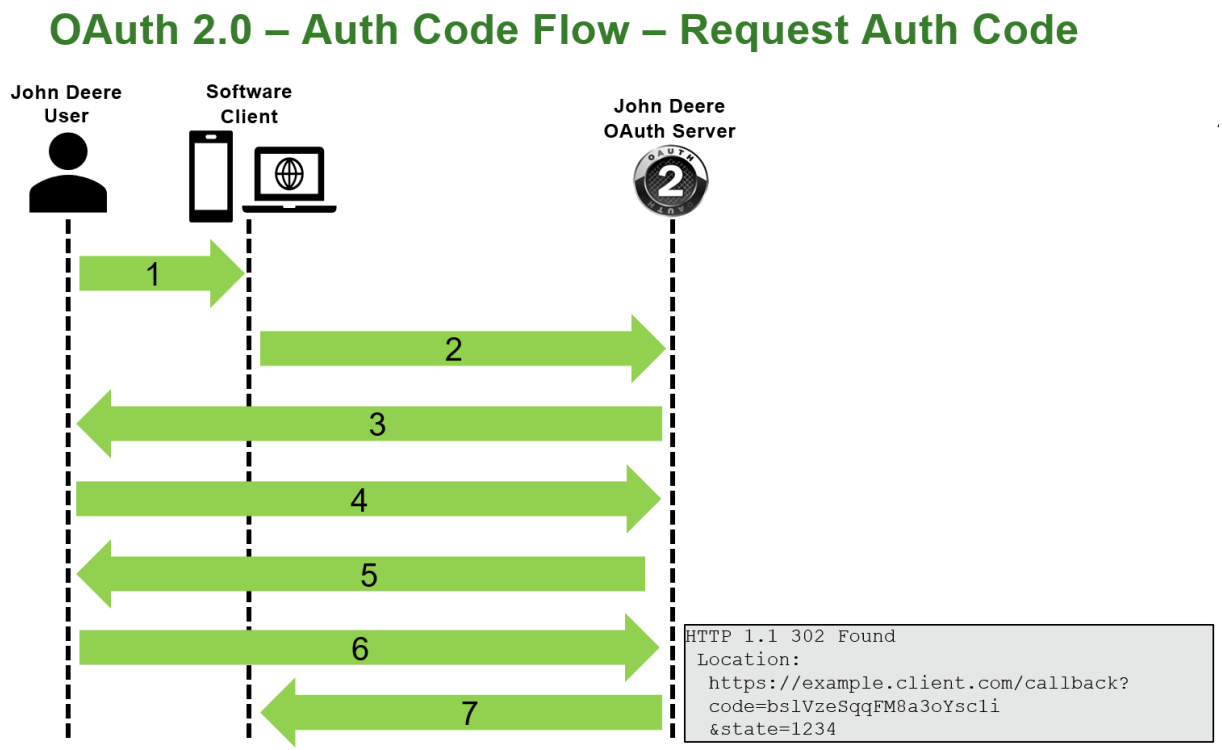


Step 5 - The customer is presented with the scope allowance screen.

OAuth 2.0 – Auth Code Flow – Request Auth Code



Steps 6 & 7 - Scope acceptance is sent back to the OAuth server, and the customer is redirected back to the client application with authorization code.



Acquire an access token:

The client requests an access token from the token server by sending an authorization grant type *authorization_code* parameter, along with the authorization code and a redirect URI. The authorization server authenticates the client and issues an access token and a refresh token (only if offline_access scope is requested). The access token expires in 12 hours.