Mamma Mia’s Accounts Payable Information System

John Wensink

MIS350 Information System Analysis and Design

Colorado State University-Global Campus

Dr. William Willette

May 12, 2019

Mamma Mia’s Pasta Company

Our company, Mamma Mia’s Pasta has grown into a sizable consumer staples company known for its dry goods, including but not limited to pasta. The company operates twenty processing facilities and logistics hubs throughout the United States, with its headquarters in Denver, Colorado. Our company uses these facilities to distribute its product nationwide to grocery stores and other retail establishments. Today, our company uses a business information system developed in-house that satisfies our needs for the most part. The two exceptions are the system for purchasing, as well as the system for accounts payable. These systems utilize MS Access, set up as a “client/server system on HP-Unix (H.P., 2019) server racks. (CSU-Global, n.d.)

**Requirements**

Mamma Mia’s physical servers are all based at the company’s Denver, Colorado headquarters. The systems perform generally well, with historical uptimes of 99.8% and downtime event duration averaging 33.5 minutes. (CSU-Global, n.d.). Because of these highly advantageous figures, we are not examining alternatives with cloud-based architectures. Instead, our company has decided a phased implementation which is iterative and incremental, will upgrade our existing network to a more robust system. This type of implementation will stand to deliver the greatest gain for the least amount of risk. My team will focus on modernizing the “Accounts Payable Information System” (APIS) adhering to agile methodologies (specifically SCRUM) with the following baseline requirements:

* The APIS must be a “standalone, independent system.” capable of disbursing payments to all vendors for goods and services.
* The APIS “will be integrated with other information systems already used in Mamma Mia.” This will be accomplished through heavy use of “Application Programming Interface” (API) infrastructure,
* Vendor invoices must include “specifications of the goods or services delivered, the prices, terms of payment, and all other related information.” (CSU-Global, n.d.)
* Ability to limit the total number of outgoing payments processed as a derivative of cash outflow forecasts.

(Khabbazi, 2015)

**Feasibility Analysis**

The goal of feasibility analysis is to determine if a project makes economic, temporal, technological, and legal sense before a large sum of manpower and capital is invested. Let’s break down the APIS feasibility analysis into its constituent categories. First the economic perspective. The Board has decided that in order to meet the changing needs of our business, an upgrade is required that will allow Mamma Mia to operate its accounts payable efficiently in today’s modern business environment. For this project to justify the cost of implementing such an upgrade, a solid return on investment must be both tangible, and realistic. One of the main benefits of this upgrade where the immediate financial benefit will be realized exists in the upgrade’s function to eliminate undercapitalization occurrences due to lack of funds to pay for invoices. The company will see benefits as soon as the upgrade is implemented by avoiding interest and fees, as well as the cost of short-term capitalization fixes such as loans.

From the scheduling perspective, I have complete faith in our IT department, as well as the “Systems Development Life-Cycle” (SDLC) we will be using. That system is SCRUM, a type of SDLC that focuses on breaking up large projects into more manageable components. SCRUM values consistent small gains, learning and failing fast, and close integration with the project’s stakeholders. We will be discussing SCRUM in more detail later in the proposal, but in my opinion, it will provide a practical roadmap in which our skilled team of systems engineers will thrive. (O’Connell, 2017)

From a legal standpoint, the landscape looks similar to the existing system. Mismanaged and overdue payments have occupied the bulk of legal’s time and our goal is to ease that pressure so that they can focus on more advantageous uses of their time like pursuing litigation against debtors who have defaulted on their payment arrangements in our Accounts Receivable department. The system upgrades will now allow the system to prioritize aged balances ahead of fresh balances to avoid vendor credit implications when we are resting vendor balances, removing time time-consuming tasks for our accounts payable legal team.

**Technological Feasibility Analysis**

From the technological standpoint, it is a bit of a relief that we will be upgrading our current network rather than migrating all data to cloud-based servers. One of the showcase technologies that will bring our company substantial efficiency will come from new API scripting. The ability to forge new robust communication links with our vendors is something that our company does not take advantage of today, which when implemented, will virtually eliminate man-hours from having to log in to each and every vendor’s website and compile data manually. Our proposed API will work to scour the information needed from these companies in the background and present the data in an organized, logical way that is intuitively displayed for the end user. This is not simple coding, and security threats here are very real, it is my recommendation that the committee allows a consulting budget to successfully integrate this technology into our APIS. (Valacich et al, 2017)

**SCRUM Structure**

We briefly touched on some of the merits of SCRUM earlier in the proposal, now let us break this broad topic down into more manageable pieces in much of the same style that SCRUM breaks down large projects into their constituent themes. One key aspect of SCRUM is a daily meeting lasting not longer than fifteen minutes. In that meeting, the SCRUM developers gather at their whiteboard, facilitated by the SCRUM Master, the development team discusses the same three items every day. What did you do yesterday? What will you do today? What obstacles are getting in your way of completing your work? The role of the SCRUM Master is to remove those obstacles, and if unable, get the “Product Owner” (P.O.) liaison involved until the problem is solved. (Trapani, 2019)

As we have said, in the Agile Methodology SCRUM, a large project is broken down into smaller, more manageable pieces. The Vision is the overarching roadmap that tells all involved where the project is headed. The Vision is about developing a “Minimum Viable Product” (MVP) for early adopters to start working on generating the highly useful QA feedback loops on which our success is hinged. Decomposing The Vision is done by breaking it down into “Themes” which are tasks that share similar characteristics. This helps us develop ideas towards a MVP efficiently and in a logical order. Themes can be decomposed into features which are smaller yet overviews of program functions such as pay invoice, or order supplies. Functions can be broken down even smaller into working packets we call User Stories. (Trapani, 2019)

A user story will follow the same general format:

* A ( ) wants to ( ) so that ( )
* An employee wants to order flour so that our stock remains full.
* A manager wants to see aged invoices so that we avoid high interest.
* A developer wants to elicit feedback from a user to improve the “Graphical User Interface” (GUI)

(McConnell et al, 2018)

User stories are of a small enough level of detail to be worked on by the SCRUM team. This is the tactical level where the day-to-day work will be accomplished. User stories are flexible until they are added to the backlog. Once the SCRUM Master, Developers, and P.O. agree on user stories as well as acceptance criteria, they can be added to the backlog of work to be completed by the developers.

**Project Structure**

SCRUM prescribes development teams of seven skilled individuals of various backgrounds to maximize communications efficiency. Tolerances of plus or minus two individuals are permitted. The use of consultants is encouraged for any sufficiently technically advanced work that is outside the development team’s skillset. The team will be lead by a SCRUM Master whose job it is to remove roadblocks for the team and facilitate the SCRUM meetings. The Product Owner liaison is a mission-critical full-time team member who is representing the interests of the stakeholders. This setup is useful for maintaining good working relationships for all involved parties and allows close coordination with project stakeholders.

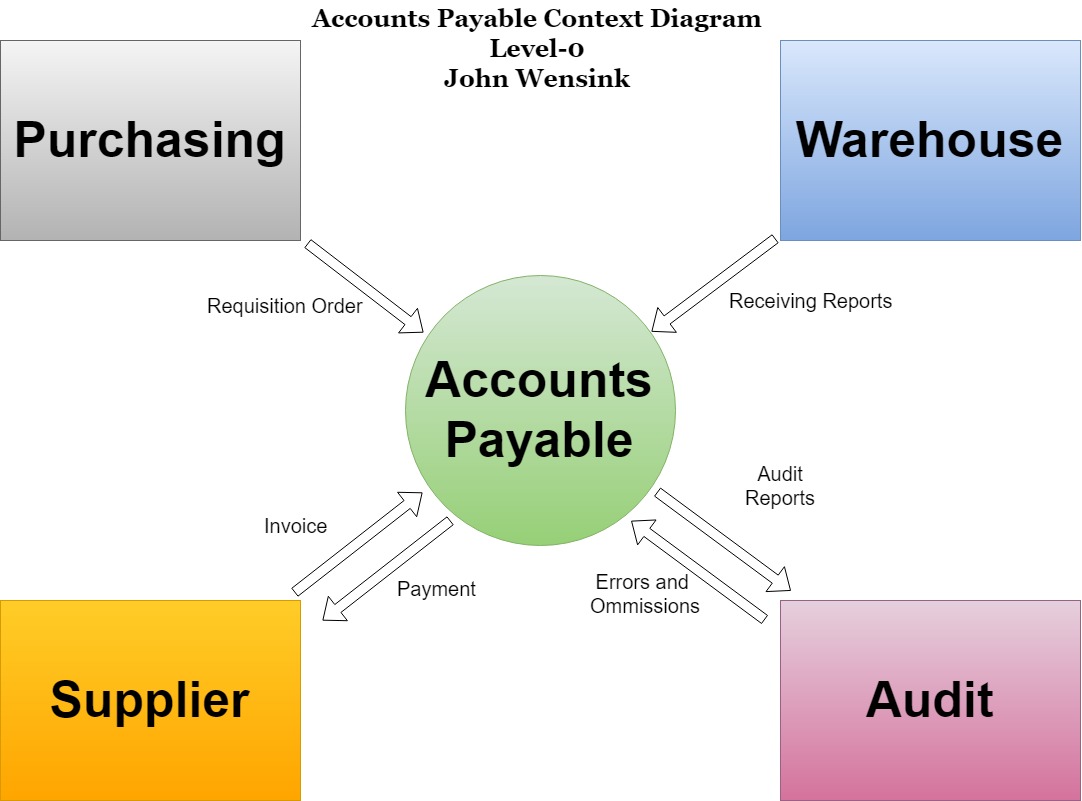
The project will run for an amount of time commensurate with the budget allotted by the Steering Committee. Time overruns will be allocated for in advance of the project starting, and work-hours will be stored in a communal time bucket which developers may draw from should the need arise. Upon completion, the end deliverable must go through network penetration testing by a team of independent information security analysts to ensure for the safety of our organization, our vendors, and our customers. This is an important step when introducing new API’s that will be communicating constantly with our vendors over a secure background connection.

**Initialization Activities**

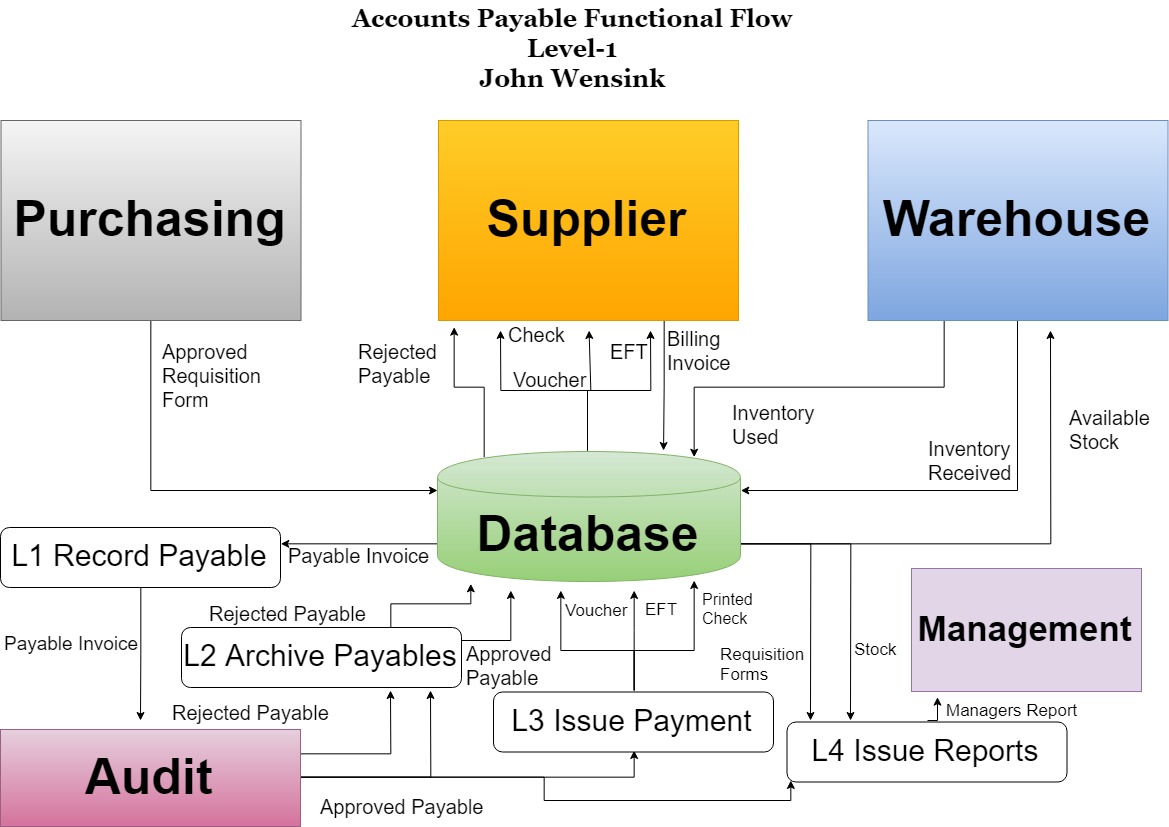
To initialize this project I request we immediately begin filling positions for project members so that these individuals can start to prepare for this project. An ideal team will consist of one expert in accounts receivable, three or four software developers, one software tester, and one elicitation analyst. We will want to get a P.O. liaison as soon as practical to start developing functions and user stories and generate a work backlog. Finally, a SCRUM Master must be selected preferably with significant systems design experience, although a can-do attitude and strong interpersonal skills should be weighed alongside resume experience. The team will need to be allocated a workspace so that they are working in close proximity of each other, ideally in the same row of an office. If this is not possible, virtualization software will allow for the acquisition of the most talented individuals for this project, regardless of geographical location.

**Proposed Intermediate Deliverables**

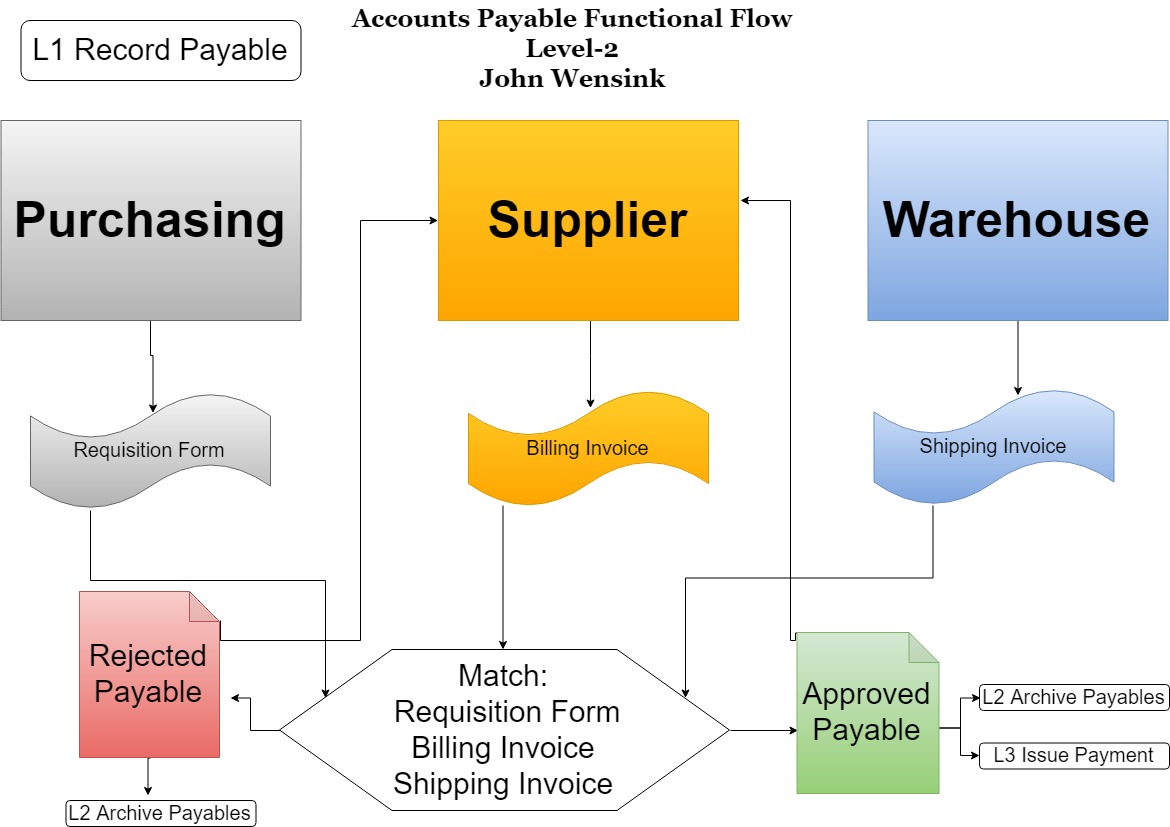
Proposed intermediate deliverables for our APIS program ought to be composed of the functions determined to be essential to the project’s success. Themes such as “Inventory Management, Account Management, Payments, and Invoices and will be further broken down into functions. In the case of Inventory Management, some useful functions might be Inventory Remaining, Days till Expiration, Cost to Resupply, and Comparison of Vendors. For the theme of Account Management, some useful functions might include Show Balance, Sort Balance By, Aged Accounts Payable, and Accrued Interest/Fees. This is where much of the work with API’s will be done. The Payments theme ought to include functions like Send Payment, Void Payment, Send Voucher, Allocate Payment, and Pay Balance. Finally, the Invoices theme would have features consisting of Sort By, Search, Print, Link to Pay, and Mark as Paid. (Valacich et al, 2017)



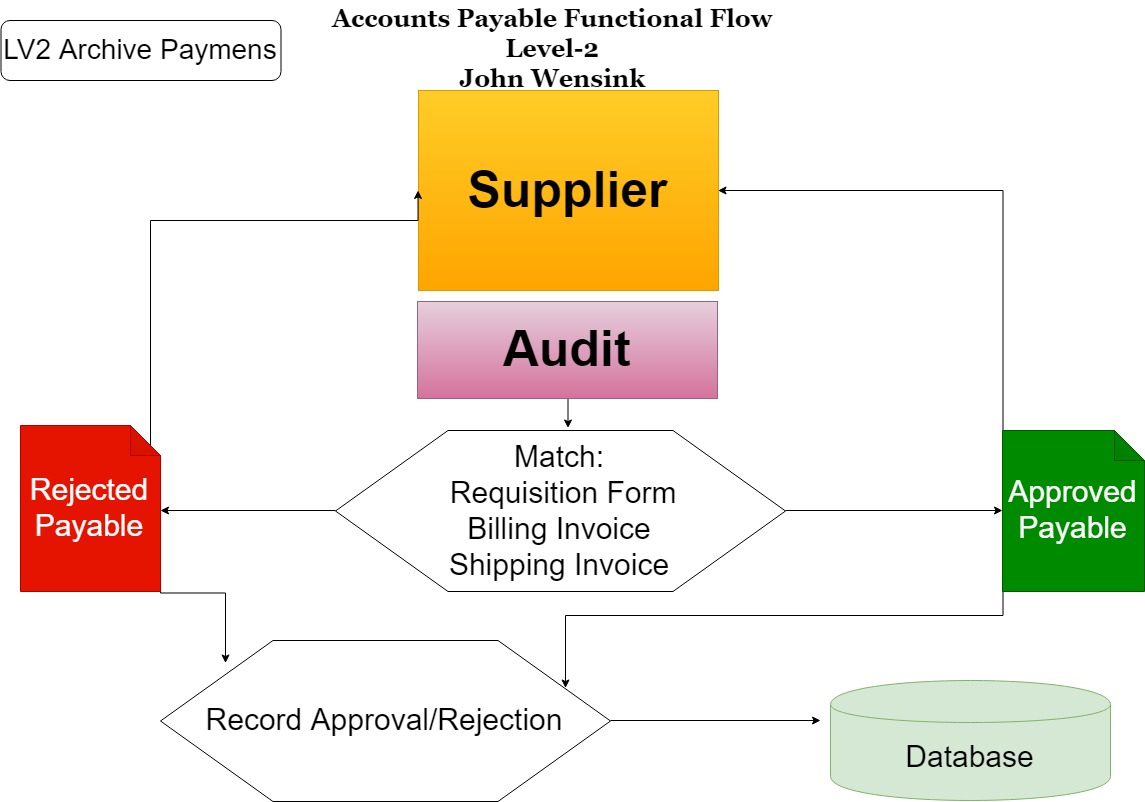
The level 0 context diagram is a broad overview of the data flowing between entities. The information will primarily interact with four main entities. Purchasing, the warehouse, our suppliers, and the audit team. This diagram details the types of information relevant to the different entities. (Ibrahim et al, 2011) (Thomas, 2016)



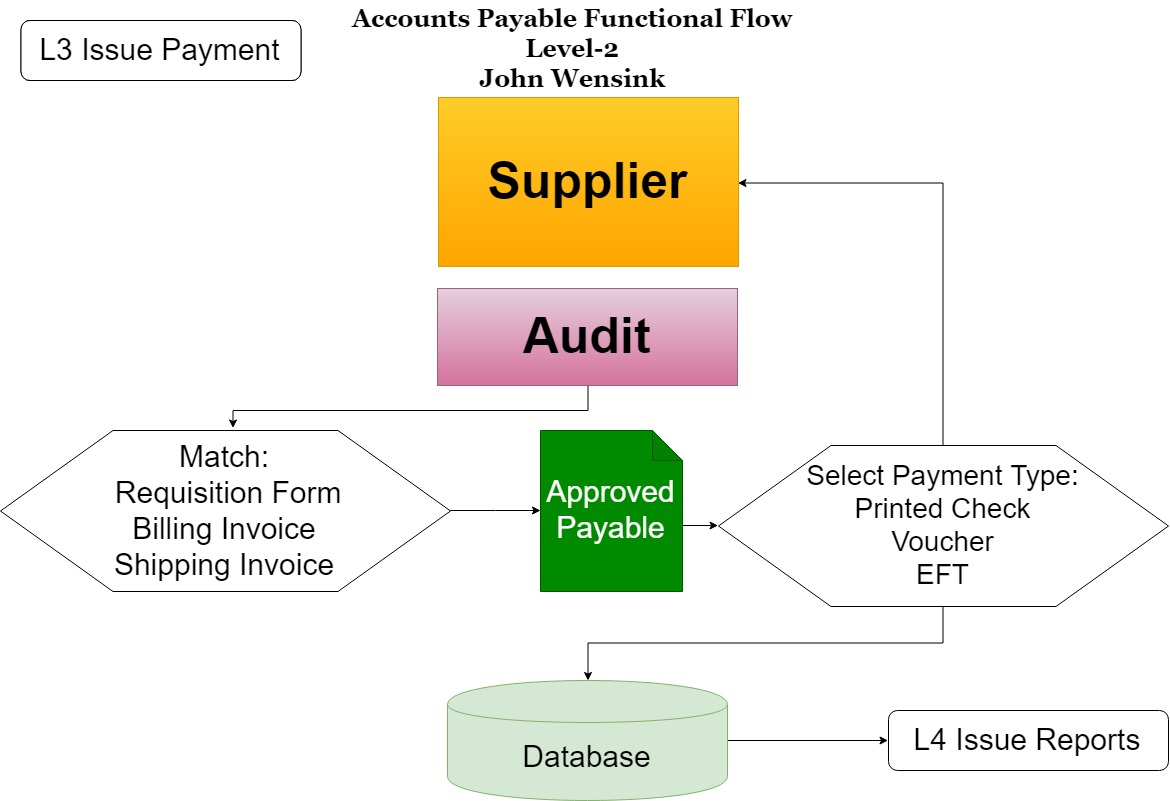
The level one functional diagram goes into more detail of the categories of data required to complete a series of tasks. Here we have four main objectives that include recording the payable, archiving the payable, issuing the payment, and generating reports. The functional flow diagram details the types of information and the direction that it travels. (Yurtoğlu, 2018)



Here, the level 2 functional flow diagram dives a bit deeper into the respective tasks from the level one functional diagram. In this diagram, the process for approving or rejecting payables is shown with more detail. (Yurtoğlu, 2018)

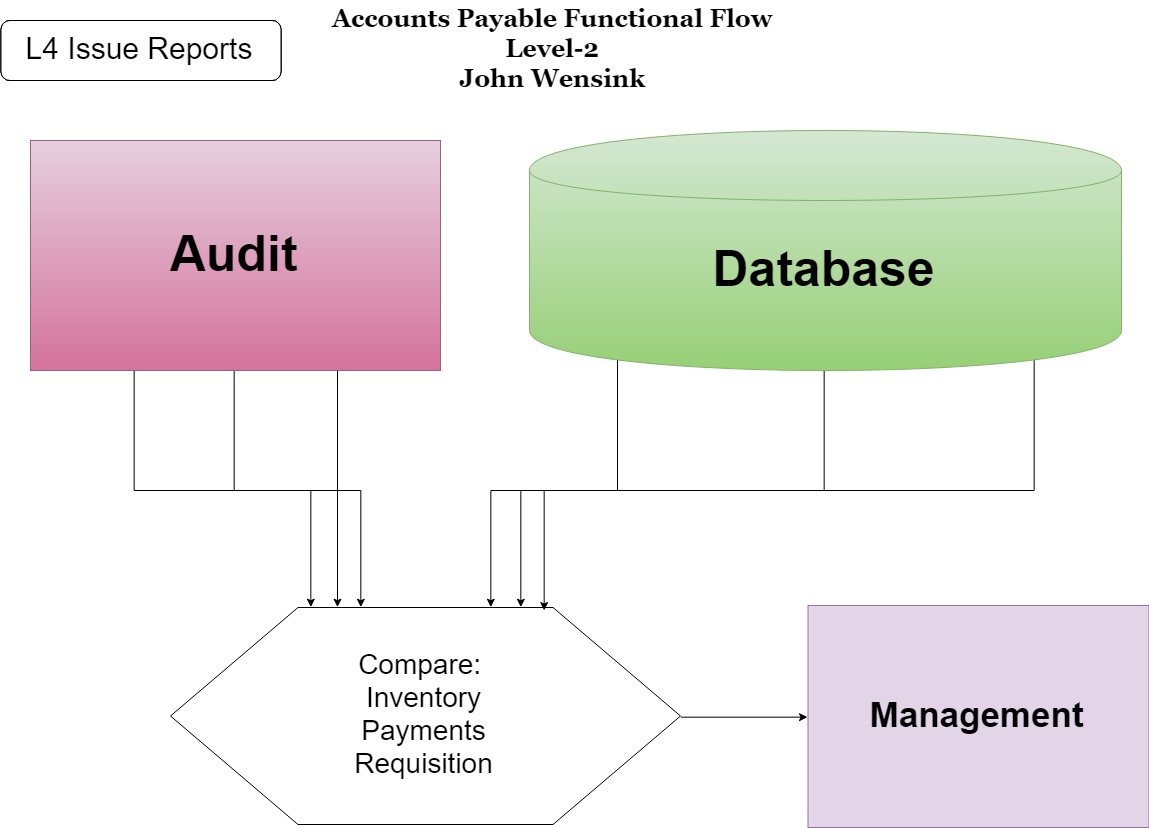


Here, the level 2 functional flow diagram dives a bit deeper into the respective tasks from the level one functional diagram. In this diagram, step two shows details on how payables will be archived. (Yurtoğlu, 2018)



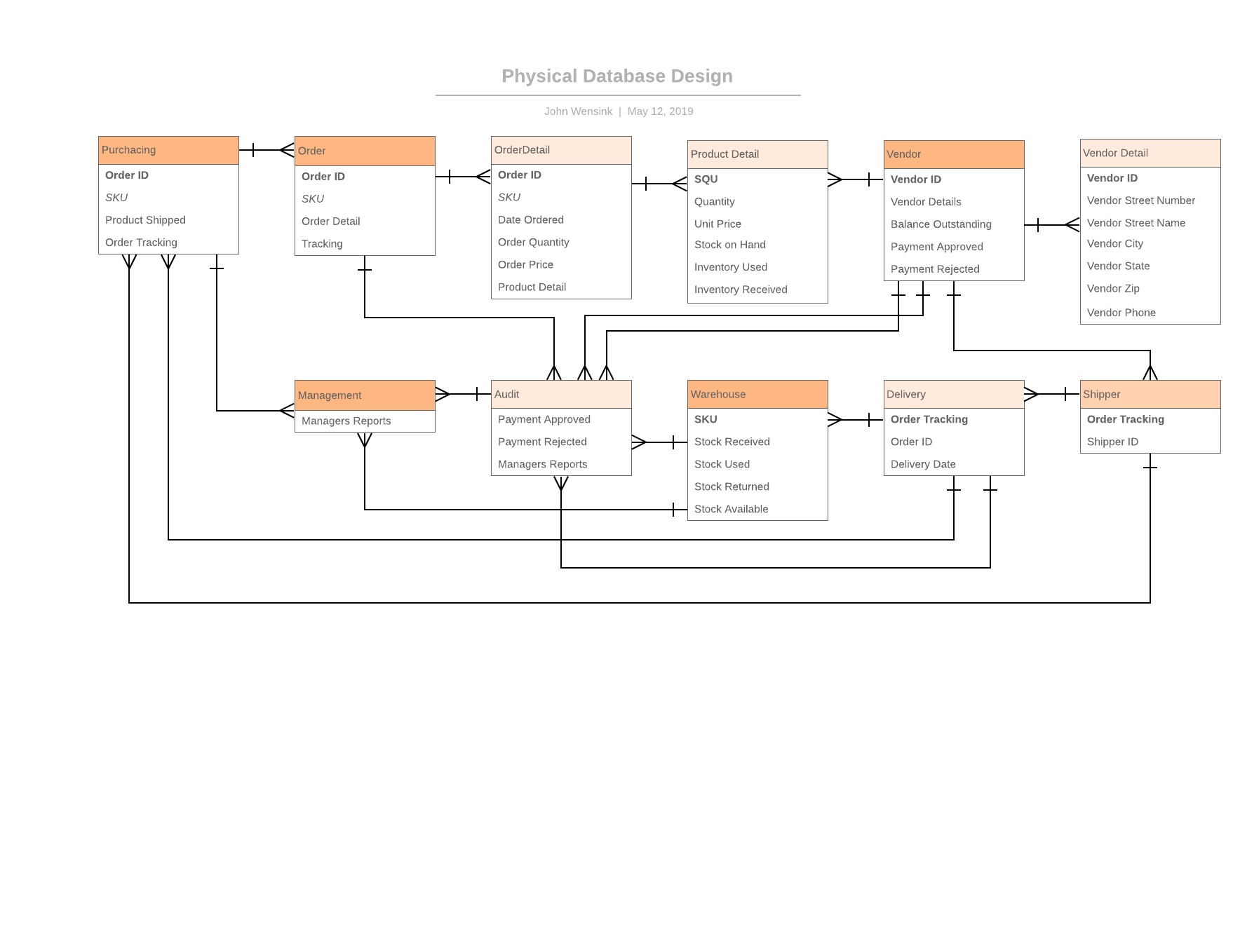
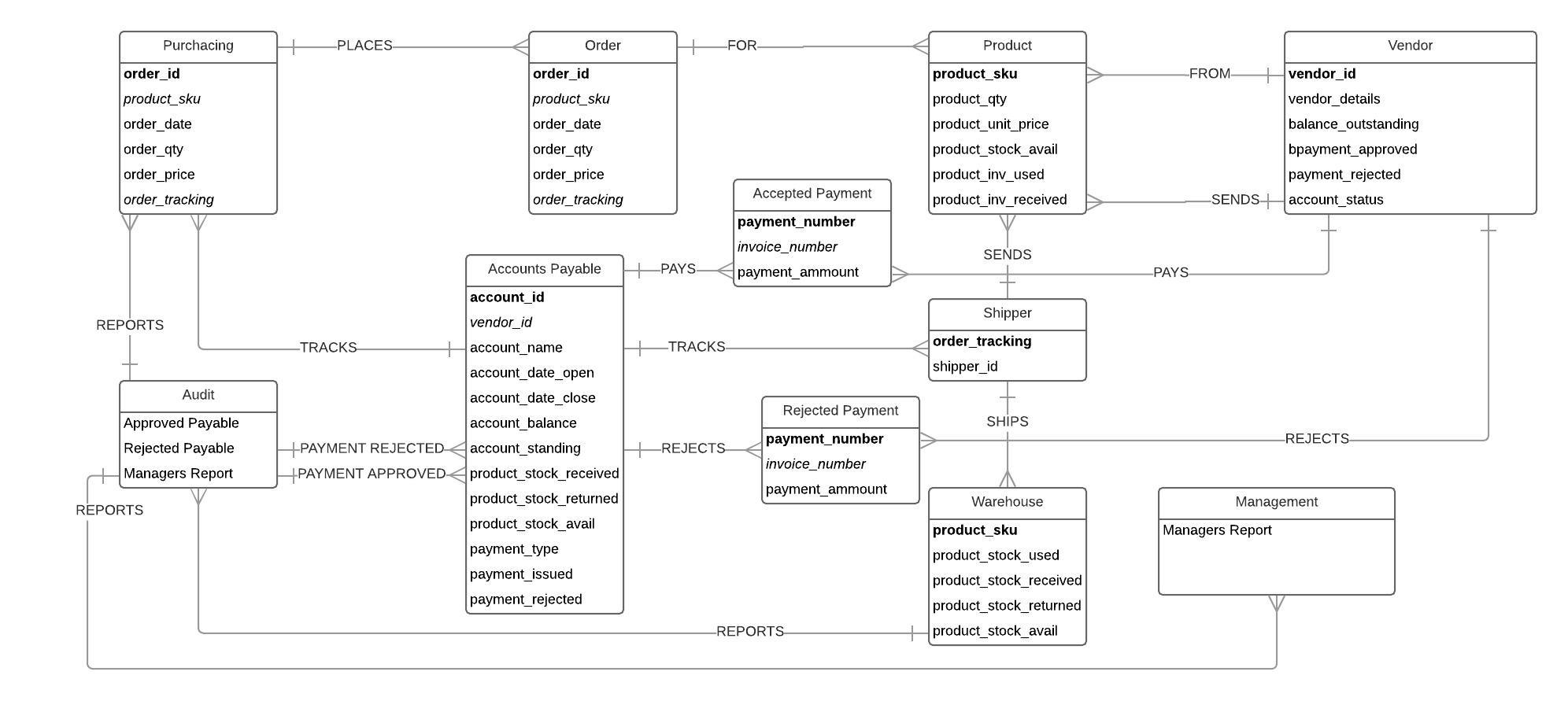
Here, the level 2 functional flow diagram dives a bit deeper into the respective tasks from the level one functional diagram. In this diagram, we take a closer look at how payments are issued.

(Yurtoğlu, 2018)



Here, the level 2 functional flow diagram dives a bit deeper into the respective tasks from the level one functional diagram. In this diagram, we see the dataflow involved in generating reports for the company’s management team. (Yurtoğlu, 2018)

“Entity-Relationship Diagram” (ERD) as well as a “Physical Database Model” (PDM.) In the ERD I have modeled the flow of a new order from start to finish. Shipping invoices are audited and either approved for payment or rejected. An inventory management system is also operated thru accounts payable. The PDM incorporates all of the elements of the ERD into tables with single value attributes paying attention to the primary keys of the ERD. The PDM also includes derived attributes from the entities they represent. The ERD has primary keys in bold and foreign keys in italic. The PDM has underlying entities highlighted in dark orange whereas the derivative attributes are highlighted in light orange. Relationship cardinality is presented as one-to-many in instances where there can be multiple instances of similar items.(Visual Paradigm, n.d.)



Accounts Payable Implementation and DevOps

As we prepare to finalize the implementation of our new Accounts Payable Information System (APIS), it is important to have a solid plan in place, and then execute that plan precisely. This transition will not happen overnight, rather, it is our aim to use a phased implementation so that we integrate the new system in slowly, paying special attention to things like compliance with user requirements, network security, and interface design. The benefit of a phased implementation is that the transition to a new system can be adopted in a series of steps where we can be sure everything is performing nominally before we begin to introduce more steps. The cornerstone of this implementation will be the accompanying training the staff as well as the managers on how to successfully implement the new information system. (Swords, 2017)

**Test Plan**

One of the benefits of a phased implementation is that problems that are discovered will be limited to a smaller cross-section of the business as a whole. Having this lower exposure to risk will allow us to test our system in a live environment, spare our company the excessive cost of full parallel implementation, as well as protect the department from major failures as we are able to isolate and address bugs. The implementation should begin with just a single vendor having access to the new system. The ideal vendor will have a small enough overall footprint that they do not represent a major supplier for our company, but with just enough volume that we are able to process enough requests that the most common bugs will begin to show themselves. The idea is that we are trying to isolate as many variables as possible until things run smoothly, and then introduce more complexity as our confidence in the system grows. When our system begins to perform well and the opportunities to address any shortcomings with the system become fewer, we can begin to raise the stakes by giving access to another small vendor so that now we can address the bugs having to do with multiple users of the system. When those bugs have been addressed we can introduce a medium-sized vendor to begin to learn how our system will scale. By the end of the implementation, our system ought to be performing well and in line with the compliance requirements set forth in earlier stages. We will continue to elicit feedback throughout the implementation and on through production use. (Valacich et al, 2017)

**Training Plan**

It is important to realize the role that the training will have on the success or failure of this project. Although we aim to make the new system as intuitive as possible, there are many specific differences from the old system and it is important for users to understand why these changes were made, how it will impact their workflow, as well as hands-on training time in a practice environment. Training should begin with the managers, who may prefer a presentation format in lieu of a simulated practice environment due to the lack of familiarity of the systems that may or may not be required for their role as managers. The first class should consist of managers and core users of the system. Once familiarized, one of the non-manager core users will be designated as a training liaison in order to assist future classes with their familiarization training. We live in a world where it is not always practical to hold classroom style training events and it is important to realize that many of our staff work remotely, it is for this reason that we make it a priority to integrate the training into a digital format that can be accessed via the web. Software such Jira by Atlassian is the standard when it comes to agile product development and bug tracking. Our company would see value in the committee investing in off the shelf collaboration software to facilitate communication, implementation, training, and production between project managers, designers, stakeholders, and users. It is this collaboration between the development team and the people administering the system that will allow us to achieve the highest level of functionality out of our new system, as well as improve future iterations. (Valacich et al, 2017)

**DevOps**

The concept of DevOps is breaking down the divide between the developers of information systems and the operations team that is tasked with managing their use. Before DevOps became popular, developers and admins rarely worked together and there existed a gap in understanding of these two very similar, yet very different sides of the information system. To successfully pull this off, we need dedicated representatives for which the the DevOps team is the main workload for those individuals. Speed and agility are two of the main benefits of a successful DevOps integration. One of the ways DevOps aims to achieve these goals is thru the automation of tasks thru specialized software. Again Atlassian leads the way with their suite of software like Bamboo, Bitbucket, and Puppet. Although not cheap, the quality of these tools will add value to our organization by allowing for a robust continuous delivery system that is intuitive and easy to use. (Meng et al, 2017)

**Implementation Strategy and Plan Using DevOps**

With the development of the beta system complete we can begin to implement the system in a phased progression that saves us both the high capital allocation requirements involved with the parallel implementation, and the risk associated with implementing the system to all users like a big-bang implementation. Implementation of DevOps, if done correctly, will ensure a feedback loop for this system so that it is constantly improving both bug resolution as well as feature enhancement. Mamma Mia needs DevOps to ensure that the investment we are making in our new accounts payable information system stands the test of time and will still be relevant years from now as the goals of the organization change, and new technologies become available. Although the costs of implementing DevOps involve significant investments in manpower, software, and space we are fortunate to have advances in technology lowering the cost of these necessities the more popular they become. With good planning, it is an investment we will only need to buy once and maintain. Without a DevOps plan it is likely that our information system would work well for a limited time, and start to become more and more irrelevant the farther we get from its implementation. (Valacich et al, 2017)

References:

CSU-Global. (n.d.). MIS350 - Portfolio Project Description. Retrieved April 14, 2019, from <https://csuglobal.instructure.com/courses/8785/modules/items/417466>

Hewlett-Packard. (2019, January). HP-UX Support Matrix. Retrieved April 14, 2019, from <https://h20195.www2.hpe.com/v2/getpdf.aspx/4AA4-7673ENW.pdf>

Ibrahim, R., & Yen, S. (2011). A Formal Model for Data Flow Diagram Rules. ARPN Journal of Systems and Software,1(2), 1-10. Retrieved April 21, 2019, from <http://scientific-journals.org/archive/vol1no2/vol1no2_3.pdf>

Khabbazi, Reza, M., & Sulaiman. (2015, June 30). Inventory System and Functionality Evaluation for Production Logistics. Retrieved April 28, 2019, from <http://www.diva-portal.org/smash/record.jsf?pid=diva2:829207&dswid=-1923>

Meng, M., Steinhardt, S., & Schubert, A. (2017). Application Programming Interface Documentation: What Do Software Developers Want? Journal of Technical Writing and Communication, 48(3), 295-330. doi:10.1177/0047281617721853

O’Connell, K. (2017, March 23). SCRUM: The Basics. Retrieved April 7, 2019, from <https://www.lynda.com/Business-Skills-tutorials/Scrum-Basics/550619-2.html?org=csuglobal.edu>

Swords, S. (2017, November 20). Software Development Project Roles and Responsibilities. Retrieved April 14, 2019, from <https://www.atlascode.com/blog/software-development-project-roles-and-responsibilities/>

Thomas, H. (2016, November 30). How to Create a Context Diagram. Retrieved April 21, 2019, from <https://www.lynda.com/Project-Management-tutorials/How-create-context-diagram/490754/561784-4.html>

Trapani, K. (2019, April 01). What is AGILE? | What is SCRUM? | Agile FAQ. Retrieved April 14, 2019, from <https://www.cprime.com/resources/what-is-agile-what-is-scrum/>

Valacich, J. S., & George, J. F. (2017). Modern Systems Analysis and Design (8 ed.). Boston: Pearson. Retrieved April 28, 2019, from <https://platform.virdocs.com/r/s/0/doc/244063/>

Visual Paradigm. (n.d.). Conceptual, Logical and Physical Data Model. Retrieved April 28, 2019, from <https://www.visual-paradigm.com/support/documents/vpuserguide/3563/3564/85378_conceptual,l.html>

Yurtoğlu, N. (2018). Importance of Data Flow Diagrams and Entity Relationships Diagrams to Data Structures Systems Design in C “A Practical Example”. History Studies International Journal of History,10(7), 241-264. doi:10.9737/hist.2018.658