# Criticality of Version Upgrades While Working with Dynamic and Sensitive Data in Customer Communication Domain Tools

**Renuka Kulkarni**

Independent Researcher, USA
Renukak12@gmail.com

**Abstract:**
Effective customer communication and dynamic customer experience are essential for success in today's business landscape. Customer communication solutions provide businesses powerful tools to manage communication efficiently, ensuring a seamless and personalized customer experience. This article explores the importance of using an up-to-date version of Customer Communication Management tools and the advantages and disadvantages of undergoing the upgrade process.

**Keywords:** Versioning, Software upgrade, Customer communication management, data, security

## Introduction

The fundamental question that often arises is what is software upgrade versus update? In the software update, minor bug fixes, fix patches, or vulnerability solutions are released that are updated in the current version of the software. In contrast, in the upgrade, the current version of the software is completely replaced with the newer version, introducing new functionalities along with fixes in the earlier version. In this article, the study of the necessity of software upgrades/and updates is discussed with respect to Customer communication solutions tools available in the market.

There are multiple benefits of keeping software up-to-date as it ensures performance improvement of the overall system, including:

- Patching security vulnerabilities
- Ensuring business compliance
- Enhancing overall performance
- Access to the latest features
- Improving compatibility with other systems

When it comes to Customer communication solutions, all of the above points are highly relevant, and there are even more compelling reasons to keep Exstream versions up to date at the customer's location:

- To keep software free from bugs
- Avoiding the risks associated with outdated versions

## Customer checklist while planning the upgrade

1. Verify that the current versions in use comply with the latest industry standards:

The first step in customer communication version upgrade planning is a detailed review of the current version of the software.to cross-check if the current version complies with industry standards and regulations. This check includes a performance check, regulatory check, and privacy criteria.

2. Assess whether the new required features the customer needs are included in the existing version. In making a decision for a version upgrade, the software team needs to analyze the customer requirements and understand the customer's exact needs to solve a business problem. Once this analysis is completed, the project team needs to explore the newer versions that have been released in recent years. Studying newer versions per the customer needs helps decide which version the system is supposed to be upgraded, making sure the upgrade effort will meet customers' needs.
3. Evaluate the costs associated with the software upgrade. Perform a detailed analysis of the various expenses related to the software upgrade, which includes licensing fees, implementation costs, and training costs to train the software team and business users. A check needs to be performed to ensure the CCM upgrade also requires the addition of servers and setups to support the functionalities. Sometimes, there are costs associated with the post-implementation for the product team for support needed during the initial phase of a post-software upgrade. After this analysis, the final analysis of long-term savings or benefits helps determine if the upgrade will increase efficiency and reduce maintenance costs.
4. Review the interfaces and compatibility of the version the organization plans to upgrade. Review the interfaces and compatibility of the version the organization plans to upgrade to.
5. Evaluation of re-platforming is necessary depending upon the organization’s business criteria. Sometimes, software upgrades lead to migration to cloud providers of different operating systems or moving from an older database system to a more modern one. It is crucial to complete the re-platforming needs analysis to ensure that leveraging new technologies enhances the CCM tool's capabilities without undergoing unplanned redevelopments.
6. Ensure the CCM product provider supports ongoing software maintenance. This is necessary because regular patches are required to address vulnerabilities and bug fixes and enhance software functionality. If a product fails to provide needed maintenance support, customers may face risks and experience decreased performance over the period.

## Architecture details of CCM

There are several ways to upgrade to a newer version of CCM software. Each way can be tailored to best-fit business needs. Generally, CCM software consists of a design database and design software installed on Windows servers or cloud-based platforms. Along with Windows software, the product also comes with a software version compatible with production and test environments on Z/Os or Linux to integrate the tool with enterprise architecture.

A design database is needed to store user access details, details of design objects, fonts, colors, etc.. In contrast, designers are the user interface tools that allow a developer and end user to design the communication using drag and drop capabilities. All users' actions are internal and stored in the database. Designer software is installed on Windows to allow users to make changes. As the developer completes the code, a special executable file is created, which is used to package the designed application to actual production or test environment code runs.

Upgrades are made on Windows software and software for the production environment.

Ways to upgrade:
The customer can decide if Windows and Environmental software needs to be upgraded or if a bug fix or a software patch can be managed by updating only installed environment software.
Options for Software upgrade:

**Complete upgrade:**
This approach requires detailed analysis for testing strategy and planning for the upgrade as the old version of CCM software was removed entirely and replaced with newer ones. In this approach, Windows and environments Software versions are upgraded to new versions of software.

**Flexible Approach:**
In this scenario, developers upgrade Windows software to a newer version of the software, and parallelly, the production environment is kept running with the old version. As the old Version is active in production, when there is a change in requirement, teams continue code changes in old and new versions of Windows software so both versions retain the same code. Once the development team completes thorough testing with the latest version, production will be shifted to the new version. This approach allows more time to test the new version of the software, and generally, there is no downtime in production.

**Environment version upgrade**
This approach mimics updates in any software where the business does not need to upgrade the entire version of Windows version software and does not need to go through the whole cycle of software upgrades. Minor fixes or new features that are introduced in the executable of software that runs on the production environment can just be upgraded to the production Environment (Linux, Z/OS, etc.)
In this approach, the team continues designing the older version, but production and testing regions run on newer software versions.

**Step-by-step upgrade**
Code chunks from each application are created and loaded into new software versions one by one. This approach also gives more testing time, but caution needs to be taken to avoid redundancy of objects as developers move code from each application to the new version. Care needs to be taken to avoid duplicating standard components. This approach is like a flexible approach, but in this case, the design software is upgraded to the new version, but the database is not upgraded.

**Steps to upgrade**
Concept validation: Create a copy of the database and software version that the organization is planning to upgrade. Create a small subset of data and perform proof of concept. Thorough testing, along with routine business activities, will help to find any potential issues. This step will significantly help in creating any critical issues and impacting business production timelines.
Backup of data: Taking a backup of the current database is helpful in case of any unforeseen issues. A backup of the database is essential for getting critical applications back in production. Also, creating quick outputs of older versions of outputs for comparison against the newer version of the software backup database will be helpful.

Appropriate set of test data: During the upgrade, using relevant data that qualifies most of the business-critical scenarios is essential. Instead of using a large volume of datasets, opt for a smaller set that still covers the most common and critical workflows. Larger datasets increase the quantity of output and make it difficult to identify any issue in testing workflows.
Maintenance check: Running database maintenance activities in the current database before beginning backup helps maintain the integrity of database objects and avoids any integrity issues.
Using comparison tools: Use comparison tools offered in the market, such as PDF Acrobat Pro, AFP Compare, or any other compassion tools that are provided in the industry. Comparing outputs builds confidence in the development and testing team about the upgrade, and it quickly locates any potential issue in the upgraded version.
Maintaining multiple versions of software: Use multiple versions of software and keep production on the earlier version of the software. Parallelly, keep upgrading the database and software, and once the appropriate time approaches, upgrade production with a newer version.

**Checklist for Testing Setup:**
Conduct a comprehensive review of the existing production setup, including software, hardware, and system interfaces. Create an exact replica of the current production environment, ensuring all resources, configurations, and dependencies are identical. Deploy and configure the latest software versions in the replicated environment to prepare for testing or migration.

1. Analyze all the critical workflows and essential business scenarios. Create test datasets for business-critical scenarios along with business-usual scenarios
2. List and evaluate all essential workflows and key business processes. Develop test datasets to cover both critical business scenarios and routine operational scenarios.
3. Determine if there are any ad hoc codes or custom scripts created to handle uncommon or exceptional scenarios
4. The upgrade must not degrade system performance and should ideally improve it. Therefore, the system's performance should be thoroughly tested

- Prepare datasets specifically for performance testing. Monitor key metrics such as speed, CPU utilization, output generation rate per second, and memory usage. Additionally, record the total time required for generating larger datasets to evaluate system efficiency
- After completing the upgrade, conduct testing using the same scenarios and datasets as before. Document all the parameters measured in the previous step to compare performance with the new version.
- Adjust configuration settings as necessary to enhance throughput by modifying parameters to optimize efficiency.
- Pinpoint the areas impacted by the newer version, including those with performance improvements. If any significant performance concerns are identified, escalate the issue by raising a technical incident with the product team- Ensure that all document generation processes (such as batch, interactive, and on-demand communications) are functioning correctly.

5. Verify that templates, logos, images, and dynamic content are being generated accurately and function as expected after the upgrade
6. Confirm that document outputs (e.g., PDF, HTML, emails, print, fax) remain consistent across the upgraded version.

7. Ensure that all security requirements and compliance needs are met.

**Pros and cons**

| Pros | Cons |
|---|---|
| New features offered in the upgraded version offer advanced processing and improved functionality. | Upgrade requires thorough technical analysis of objects and workflows; failure to complete analysis can lead to production delays and more extended production downtime. |
| All the bugs and issues in older versions are generally fixed in the upgraded version. | There is a cost involved with version upgrades. The project needs to consider the cost of upgrades along with maintaining older versions of software for a smooth production move. |
| New software is an industry complaint offering the latest security patch | Upgrades can be complex, and that might create the need for technical training and can add further cost to version upgrades |
| Technical support provided by Product covers various areas, including version support, support for upgrades, prost, and release issues. | The project team needs to consider the cost involved in the testing effort, along with communication testing and the cost of testing all integrations. |

**Conclusion:**

Customer communication is a critical domain for any organization. Keeping the version up-to-date is vital for various reasons. Newer versions offer improved capabilities, enabling businesses to deliver communication in an effective way. As the vulnerabilities are addressed, data security and security standards are achieved. It allows seamless interaction with customers, achieving higher customer satisfaction rates. Upgrade keeps business function relevant by supporting modern communication channels. Overall, upgrading the CCM version to a higher version involves cost associated with it, but in return, it offers advanced functionalities and gives access to newer developments in software, providing modern features, and the organization can create more interactive and highly dynamic communications at the same initial planning holds important as detailed analysis and testing strategies help smooth transitions between the version reducing unforeseen costs with production issues.

Regular upgrades help maintain system reliability and ensure your communication tools continue to meet the needs of your business and customers effectively.

**References**

1. “It’s Technically Easy to Upgrade to the Latest Version of Exstream”.opentext.https://blogs.opentext.com/technically-easy-upgrade-latest-version-exstream/.Apr 11, 2016
2. “What to consider before upgrading to OpenText Exstream”.ecodocx.https://ecodocx.com/blog/upgrading-opentext-exstream/.Jul 3.2018

3. Kami Vaniea, Yasmeen Rashidi.”The process of updating software”.researchgate.https://www.researchgate.net/publication/301935796_Tales_of_Software_Updates_The_process_of_updating_software,May 2016
4. Parth Patel. ”Why You Need to Upgrade Software? Explaining With the .NET Story”.cmarix.https://www.cmarix.com/blog/why-you-need-to-upgrade-software-explaining-with-the-net-story/.Jul 4.2019
5. Gareth Griffiths,”Overlooking the importance of updates and upgrades”,techradar,https://www.techradar.com/news/overlooking-the-importance-of-updates-and-upgrades,Jul,2,2018