Avoiding Technical Debt in Elections:
Get Modernized and Stay Modernized

MTX | Election and Voter Registration Solutions

David Butter | Chief of Client Outcomes
Ed Smith | Vice President, Election Solutions

MTX Group, Inc.
6303 Cowboys Way, Ste. 400
Frisco, Texas 75034
mtxb2b.com | https://www.mtxb2b.com/s/election-microsite | elections@mtxb2b

What is Technical Debt?

Technical debt, also known as *code debt* or *design debt*, occurs when decisions are made causing development teams to prioritize speed of release over the production of the most well-designed code. This process can unfortunately lead to the accumulation of a messy, inefficient, and fragile codebase that makes future modifications harder and more costly as it drains a team's productivity and resources. Technical debt is often the result of using quick fixes, patches, and taking shortcuts made to meet immediate demands rather than the implementation of full-scale solutions. While these shortcuts offer short-term gains, they lead to complexity and higher maintenance over time.

Why is Technical Debt Incurred?
An organization takes on technical debt for reasons similar to why they might take on financial debt. For example, something is needed now - or imminently - that the organization does not have the cashflow to pay for in full. Thus, they “borrow” to get what they have deemed necessary. With software solutions, this usually means making coding or design decisions that are less than optimal – or that the decision makers know will need to be addressed and updated in the future in order to get what they want or need into production sooner.

Following are some of the most common causes of technical debt:

- **Rushed Delivery**: Cutting corners to meet tight deadlines often leads to errors.
- **Neglected Refactoring**: Skipping necessary improvements leads to brittle code.
- **Insufficient Testing**: Not properly testing the code leads to more bugs over time.
- **Overcomplicated Solutions**: Introducing unnecessary complexity can be problematic as complex code tends to break more often.
- **Outdated Dependencies**: Relying on obsolete libraries or frameworks creates issues.
- **Poor Documentation**: Lack of clear comments or guidelines makes future work slower and more error-prone.

How can Technical Debt be Minimized?
Some amount of technical debt is considered inevitable, but organizations must strike a balance between business goals and development decisions, lest they be considered “penny-wise and pound-foolish.” Client deadlines and speed-to-market decisions are paramount in the marketplace, as is superior quality and functionality of solutions. This balance is crucial as a solution development team that never makes anything other than the most optimal choices in their code is going to be hard-pressed to deploy their solutions in a regular or timely manner. A financial analogy comes into play again, as debt must be used responsibly. One takes on debt understanding that it will need to be repaid. Continuously piling on new debt without ever reducing existing balances is going to cause a variety of challenges – if not actual ruin – at some point in the future.

For many organizations, the big picture goal is to move money away from legacy tools and processes toward more innovative and transformational work. Other related goals may include a better user experience - for example, one with an updated SaaS application as opposed to a legacy one, a more agile IT infrastructure, and cost savings. One pain point is that legacy systems and processes tied up with technical debt often involve a complicated web of interdependencies, of both tools and policies. This complex web required methodical untangling.

Here are some ways to address technical debt:

- **Assess Existing Technical Debt**: Cataloging technical debt is critical. Organizational leaders should establish a list of their critical technical debt. They should know the business impact of that debt and they should have a process for addressing it.
- **Prioritize Code Quality**: Focus on code quality from the start, and take the time to review all areas of a project in order to reduce duplication and adhere to naming conventions. Set coding standards and establish best practices early. Carefully plan trade-offs between delivery time and code quality.
- **Automate Testing:** Write scripts that automatically test the software system to ensure code quality is maintained over time. Perform unit tests by checking individual components for correctness. Conduct integration tests that ensure different modules work together and undergo end-to-end tests that verify the full system functions as expected.
- **Manage Dependencies**: Regularly update third-party libraries to avoid security risks and compatibility issues. Staying current prevents more significant problems from arising when older libraries become obsolete.
- **Conduct Robust Code Peer Reviews**: Regular peer reviews catch potential issues early and ensure best practices are followed. This collaboration encourages development team members to share knowledge and work together to maintain a high-quality codebase.
- **Maintain Clear Documentation**: Write concise, helpful comments and keep documentation up to date. Well-documented code makes it easier for other developers - or your future self - to understand and maintain the solution.
- **Plan for Technical Debt:** After assessing existing technical debt, identify signs of debt, measure the time needed to reduce it, and create a plan. Debt can't successfully be managed if it's not measured, and thus, organizations need to get better at identifying, tracking, and planning for technical debt.

Avoiding Technical Debt in Elections: Get Modernized and Stay Modernized

We have entered a new era in election solutions. Implementing a modernized solutions platform is not just a solution for managing elections—it is a vision for operating that results in building the future of secure, flexible, and accessible election solutions specific to the needs and requirements for individual states and election jurisdictions. The ability of states and election jurisdictions to **get modernized** and continuously adapt ensures that election management processes remain efficient, secure, and voter-centric—and thus upholding the highest standards of electoral integrity and trust.

Incorporating a Visionary Solution for Secure and Voter-Centric Election Management
In an era where public trust, electoral integrity, and secure citizen engagement are paramount, election jurisdictions are challenged to modernize and adapt to evolving state and federal election requirements. These challenges are not just operational, but foundational. They are driven by the need to uphold democratic processes while fostering voter accessibility and trust. The need for a transformative solution emerges for election management, offering innovation that not only enables states and local election jurisdictions to **get modernized,** but ensures they **stay modernized**, evolving in step with the rapidly changing electoral landscape.

Navigating the Rocks and Shoals of Electoral Change
Election jurisdictions must balance the growing complexities of regulatory compliance, evolving voter expectations, and the need for transparency—all while responding to budgetary constraints and shifting political priorities. Elections management solutions engineered for this dynamic environment deliver rapid deployment and the agility to adapt quickly. Their flexibility ensures election management solutions can address new laws, emerging technologies, and the unique demands of each election cycle with precision and confidence, allowing jurisdictions to modernize.

Organizational Evolution and Technological Advancement
As election offices adapt to changes in staffing, organizational structure, and advancements in technology, election solutions providers can enable a robust yet adaptable foundation by implementing low-code no-code capabilities which empower state and local election administrators to innovate without requiring extensive technical expertise. This allows for quick responses to changes such as new voter registration laws, redistricting, and cybersecurity mandates. Judicious platform selection allows integration of emerging technologies which ensures election management solutions remain resilient, secure, and aligned with the cadence of change, supporting both modernization and sustained evolution.

Building Trust Through Continuous Modernization
In today's rapidly changing environment, trust in the electoral process is critical. Thus, election solutions providers must offer election officials the tools to **get modernized and stay modernized**, reducing inefficiencies and vulnerabilities while maintaining the integrity of voter data. With secure cloud-based infrastructure, FedRAMP certification, and regular platform updates, the right platform ensures election solutions are always compliant with rigorous security standards and prepared for emerging threats.

Empowering Democratic Engagement: Not all Platforms are Created Equal
A comprehensive and flexible election management platform goes beyond traditional legacy election systems, enabling election jurisdictions to transform the voting experience. By leveraging built-in analytics and reporting capabilities, election officials can monitor real-time voter engagement and system performance, addressing potential issues proactively to ensure smooth operations on, before, and after Election Day. This adaptability ensures that as voter expectations evolve, election management solutions keep pace, offering continuity and trust. **By offering platform choice, election solution providers can help election jurisdictions avoid costly technical debt.**

Bibliography

- Casey, Kevin. *"How to explain technical debt in plain English",* The Enterprisers Project. June 23, 2020. https://enterprisersproject.com/article/2020/6/technical-debt-explained-plain-english
- Layton, Mark. "*The real cost of technical debt*", Agile Techniques Blog from Platinum Edge. Feb 26, 2019. https://platinumedge.com/the-real-cost-of-technical-debt
- News & Insights Team. "*New Salesforce Innovations Empower Creators to Build Apps with Low Code on a Single Platform*", Salesforce Website. June 23, 2021. https://www.salesforce.com/news/press-releases/2021/06/23/salesforce-platform-innovations/
- Pratt, Mary K. "*5 Tips for Tackling Technical Debt*", CIO. April 12, 2023. https://www.cio.com/article/472768/5-tips-for-tackling-technical-debt.html
- Shahrabi, Shahriar. "*Why Avoiding Technical Debt Might Be Your Biggest Mistake: Embrace Progress, Not Perfection*", Medium. Nov 4, 2024. https://shahriyarshahrabi.medium.com/why-avoiding-technical-debt-might-be-your-biggest-mistake-embrace-progress-not-perfection-b13a3fe2c91f
- Sugosh, Mrina. "10 Tips to Manage Technical Debt", Built In. Oct 29, 2024. https://builtin.com/articles/tips-to-manage-technical-debt