A HEALTHCARE DATA ANALYTICS MANIFESTO

Healthcare organizations have access to unprecedented levels of data and processing power, and urgent and complex questions to answer. But despite the industry's enormous investments in data and analytics, most organizations still struggle to bring their data to bear on even their most important decisions.

Organizations can achieve their innovation and transformation goals by using tools that balance structure and flexibility, and by emphasizing rapid, iterative development with deep collaboration between technical staff and subject matter experts.

STRUCTURED YET FLEXIBLE

WHAT DOESN'T WORK

DOESN'T WORK: Analytics software with immutable, one-size-fits-all logic that produces beautiful dashboards but whose results don't stand up to scrutiny.

DOESN'T WORK: Developers writing novel SQL for each new request, often independently and from scratch, with no tools beyond their favorite query editor.

WHAT DOES WORK

DOES WORK: Analytic tools with sophisticated built-in logic that still allow the tailoring necessary to accommodate local data idiosyncrasies and business rules.

DOES WORK: A flexible, object-oriented "living data model" that grows with an organization's analytic needs, allowing teams to easily reuse and extend prior work.

Historically, organizations have had only two available approaches to produce the analytics that answer their business questions.

The first is to buy a vendor's analytics product. The promise is that the software will automate and fool-proof the process of generating complicated analytics. But the software's hard-coded logic inevitably fails to handle the idiosyncrasies and exception cases present in every organization. End users lose confidence in the results, and the software fails to deliver business value.

The second path solves the problem of inflexibility by making each request a custom job—designed and implemented, usually from scratch, by a team of analysts and developers. Such projects are typically one-offs, with the code packaged into one or more stored procedures that only the original developers really understand. But hand-crafted customization is resource intensive, takes a long time, and is hard to maintain.

Between these two extremes is a better approach: software tools that provide just the right amount of structure and automation, leaving room for the ingenuity and flexibility of a development team and allowing for as much customization as needed.

COLLABORATIVE AND ITERATIVE

WHAT DOESN'T WORK

DOESN'T WORK: Assembling a team of analysts and subject matter experts to carefully craft requirements into a report request, then making it very hard to amend those requirements later.

DOESN'T WORK: Instructing developers to produce a deliverable that meets the needs of end users using only those requirements, on the first try, without any mid-project interactions with subject matter experts.

WHAT DOES WORK

DOES WORK: An expectation that requirements will need to change as developers encounter data idiosyncrasies and unforeseen corner cases, and as end users learn from these to modify the details of their analytical question.

DOES WORK: A culture of collaboration between technical staff and subject matter experts, embodied by a project plan that includes multiple cycles of iterative development, data review, and refinement.

Some custom development will always be necessary, but traditional software development workflows are inflexible and inefficient. In particular, waterfall-style project structures, in which revisiting decisions made in prior stages is discouraged, nearly always yield inaccurate or irrelevant final work products. Subject matter experts need a chance to see the data presented in an understandable way and amend their requirements, and developers need a chance to learn the subject matter and refine their implementation.

In face-to-face sessions, developers and analysts should present detailed, case-level data to subject matter experts and end users, highlighting what they see as suspicious or unexpected, and participating fully in any resulting adjustment to requirements. A simple deliverable might need one or two sessions, while a complex one might need several more, but the sessions should be scheduled days, not weeks, apart. This aggressive pace keeps the deliverable front of mind and saves developers from wasting time going down a wrong path. The idea is to quickly find mistakes or oversights, in either the requirements or the implementation.

Organizations should establish a culture of collaboration and respect between technical teams and business or clinical staff, and emphasize intellectual courage, humility, and a willingness to learn in hiring and professional development. In addition, software tools that facilitate rapid prototyping of business or clinical logic, manage metadata (e.g., reference data and field definitions), and securely share case-level data help by letting developers spend more time working at the top of their license.

Ursa Health