

Two major realms of application systems exist in the standard and quasi-standard *networked* world today: one is the traditional host based realm which is best represented by IBM mainframes that are interconnected via SNA (Systems Network Architecture) - hereafter referred to as NLS (= Networked Legacy Systems). The other is the emerging realm of dynamic/interactive web applications which are based on client/server architectures that are linked via standard Internet protocols (hereafter referred to as WAppS (Web based Application Systems) and WDBs (Web based Data Bases).

Applications of both realms perform transaction processing which by definition includes database access and update. Beyond transaction processing, Enterprise Intelligence Systems (EIS) access databases in both realms.

Issue 1: The Weight of Sunk Money. The accrued dollar investment in NLS has arrived at such immense amounts, and more than that, the dependence on the respective infrastructures has evolved to such enormous degree, that any quest for or attempt to replacing NLS applications and databases has proved to fall by the wayside. Decision- makers are averse to the risk involved and are more willing to swallow the bitter pill of huge running costs. There will be no short or medium replacement for NLS, maybe even for a very long-term.

Issue 2: Unclean Programs/Lack of Documentation. Part of the problem of being locked into NLS (apart from the proprietary nature of the systems) stems from the way, NLS applications have been programmed. NLS applications frequently lack proper documentation, they are coded in anything but object oriented programming languages, so these applications overwhelmingly handle everything in a single program: the user interface (UI), the business logic (BL), and the database access (DBA) which is -by the way- the deeper root of the Y2K problem. Regarding their lifespan these applications are almost unmaintainable.

Issue 3: The Need for Interoperability. Since most relevant corporate data (and presumably most relevant government data also) reside on NLS(1), the question arises how both realms are or can be connected in a way that at least WAppS may have access to both NLS applications and databases.

Issue 4: Here We Go Again. Applying lessons learned from the NLS experience means to (1) prefer object orientation in all WAppS development projects which also includes proper documentation throughout all project levels, (2) strictly restrict developments to open, nonproprietary systems (even at the expense of missing a fancy feature for a while), (3) understand the "application program proliferation law" that says application programs and their attached structures (database, UIs) last longer than ever expected. When looking at many WAppS development projects, however, the lessons learned are being ignored; many WAppS do not comply with any of the rules outlined above. Linking such unclean WAppS into NLS will create tremendous ramifications of an old problem, needless to say that they will also lead to prohibitive running costs.

Issue 5. The Threat of Stall. While the demand for decision oriented "actionable information" and for decision process relevant knowledge obtainable from IT systems among managers and other user is ever growing, the integration of (unclean) NLS and unclean WAppS will make reaching this goal ever more challenging if not impossible.

(1) See Internetweek, July 20, 1998, Legacy Migration: Scaling to IP