EMERGING PRINCIPLES FOR GUERRILLA ANALYTICS DEVELOPMENT

Research-in-Progress

Enda Ridge
Data Insight Services
Risk Consulting
KPMG
enda.ridge@kpmg.co.uk

Edward Curry
Digital Enterprise Research Institute
National University of Ireland, Galway
ed.curry@deri.org

Abstract

Analytics projects come in many forms, from large-scale multi-year projects to projects with small teams lasting just a few weeks. There is a particular type of analytics project identified by some unique challenges. A team is assembled for the purposes of the project and so team members have not worked together before. The project is short term so there is little opportunity to build capability. Work is often done on client systems requiring the use of limited and perhaps unfamiliar tools. Deadlines are daily or weekly and the requirements can shift repeatedly. Outputs produced in these circumstances will be subject to audit and an expectation of full reproducibility.

These are 'guerrilla analytics' projects. They necessitate a versatile and fast moving analytics team that can achieve quick analytics wins against a large data challenge using lightweight processes and tools. The unique challenges of guerrilla analytics necessitate a particular type of data analytics development process.

This paper presents research in progress towards identifying a set of development principles for fast paced guerrilla analytics project environments. The paper's principles cover 4 areas. Data Manipulation principles describe the environment and common services needed by a guerrilla analytics team. Data Provenance principles describe how data should be logged, separated and version controlled. Coding and Testing principles describe how code should be structured and outputs tested. All these principles focus on lightweight processes for overcoming the challenges of a guerrilla analytics project environment while meeting the guerrilla analytics requirement of auditability and reproducibility.

Keywords: guerrilla analytics, data analytics, agile development

Introduction

In a 1958 article, Hans Peter Luhn from IBM used the term Business Intelligence (BI) to define "the ability to apprehend the interrelationships of presented facts in such a way as to guide action towards a desired goal" (Luhn 1958). Since Luhn's article the field has evolved into a set of methodologies, processes, architectures, and technologies that transform raw data into meaningful and useful information used to enable more effective strategic, tactical, and operational insights and decision-making (Lavalle et al. 2011). The core of modern BI is the data analytics that transforms an organization's data into actionable insight.

Analytics projects come in many forms, from large-scale multi-year projects to ones with small teams lasting for just a few weeks. There is a particular type of analytics project identified by some unique challenges. In these projects, a team is assembled for the purposes of the project and so team members will not have worked together before. The project is short term so there is little opportunity to build team capability. Work is often done on client systems requiring the use of limited and perhaps unfamiliar tools. Deadlines are daily or weekly and the requirements can shift repeatedly. Outputs produced in these circumstances will be subject to audit and an expectation of full reproducibility.

These are 'guerrilla analytics' projects. They necessitate a versatile and fast moving analytics team that can achieve quick analytics wins against a large data challenge using lightweight processes and tools.

The unique challenges of guerrilla analytics necessitate a particular type of data analytics development process. This paper presents research in progress towards identifying a set of development principles for conducting data analytics in fast paced guerrilla analytics project environments.

Guerrilla Analytics

Examples

To better understand guerrilla analytics, consider some typical projects that prompted the development of the principles introduced in this paper.

Forensic Investigations (Westphal 2009) **and Financial Remediation** (Culp et al. 2012). These projects aim to determine how a fraud was committed against a client by a customer or by the client's employees. They cover situations where a regulation was broken such as dealing with countries or individuals on a recognized sanctions list or committing bribery. Outputs will contribute to a legal case, employee dismissal or a government commissioned report and so must be fully auditable and will come under intense scrutiny. There is usually intense pressure to come to an answer quickly.

Data journalism (Rogers 2012)**.** These projects are reactive in that to be relevant they must respond to the news, which is inherently unpredictable. Deadlines are usually for daily or weekly publications. Data is drawn from research, government publications, and whistleblower data such as Wikileaks[1]. Reporting must be clear and intuitive since the audiences are not technical or legal experts in the subject matter.

Distinguishing Characteristics

In analytics projects, the objective is to identify and acquire the relevant data, understand the data and mine it to produce insights. These projects, while challenging, are well understood and have clear project phases. Consider the following characteristics that in combination create a guerrilla analytics scenario.

- **Working on client systems**: the project will often necessitate that analytics work be conducted on client systems as opposed to the analytics team's servers. Tools available will be limited to those licensed by the client. A guerrilla analytics team needs simple processes that can be implemented with a minimum of tools.
- **Client mobilization under tight deadlines**: The team is often working with client data and it can be difficult for a client to quickly mobilize the appropriate people to provide the business and systems

[1] http://wikileaks.org/

knowledge to support the project. This leads to delays in finding the correct data and interpreting its business context. However guerrilla analytics projects cannot be delayed. Guerrilla analytics requires incremental interaction processes with client personnel to support development and data processes on fast moving and incomplete data.

- **Undocumented data:** much of the data is undocumented, or documentation is out of date. The guerrilla analytics team must be adept at reverse engineering data so the project can progress while the appropriate data and documentation are identified in parallel.
- **Diverse interacting teams generating data:** the analytics team will be complimented by domain experts in fraud, risk, accounting, and other areas. These team members are used to generating data using their own tools and work processes, but their outputs must be incorporated into the analytics team's data repository. However, there is never time to generate controlled data workflows for this purpose. Guerrilla analytics requires lightweight processes to control the workflow and quality of human generated data.
- **Resourcing & Up-skilling**: guerrilla data analytics teams have a rare skills mix. They must be familiar with scripting, data manipulation, visualization techniques and machine learning (Kohavi et al. 2002). Furthermore, the team must understand software engineering techniques and tools to cope with the pace of the project while maintaining team coordination. In the cases we examined, the majority of data analysts have backgrounds in statistics, mathematics or engineering but do not have good awareness of software engineering tools and processes. As teams typically only exist for the duration of the short lived project, there is little time to up skill the team. Guerrilla Analytics processes and workflows must be lightweight and easy for team members to understand and use.
- **Evolving requirements under tight deadlines:** The analytics team must produce deliverables with deadlines of the order of a day and changing requirements while maintaining high quality, tested outputs and that meet the highest standards of auditability. The guerrilla analytics team must have processes to consolidate its data understanding for ease of re-use and consistency of deliverables.
- **Audit and reproducibility.** It is often a regulatory requirement that results be audited. Results presented in a court of law must be forensically sound. An auditor of analytics work will expect:
    - full traceability of all data received with documented and reproducible checks that the data was complete and uncorrupted.
    - full traceability and documentation of all business rules implemented with clear demonstration of how application of those rules affected the population of data being analyzed.
    - full documentation and reproducibility of key project deliverables.

Before presenting a set of principles that address these challenges, we first look at the most appropriate existing development approach from software engineering.

Development Approaches and fit for Guerrilla Analytics

Agile Software Development (Cohn 2010; Martin 2003) is a software development approach that can be characterized by allowing requirements and solutions to evolve through collaboration between a customer and self-organizing cross-functional teams. Agile promotes adaptive planning, evolutionary development, a time boxed and iterative approach to delivery, and encourages rapid flexible response to change. These concepts are encapsulated in the Agile manifesto (James Shore and Warden 2007) which values: (1) individuals and interactions over processes and tools, (2) working software over comprehensive documentation, (3) customer collaboration over contract negotiation and (4) responding to change over following a plan. Agile has many virtues for analytics development. However there are several areas in which Agile software development does not quite fit the needs of guerrilla analytics development.

- Agile software development's time boxed delivery is usually of the order of weeks (Cohn 2011). Deliverables in guerrilla analytics projects by contrast are usually required within several days.
- Guerrilla analytics projects require that some types of detailed documentation be produced for audit. Working code is still essential but detailed supporting documentation cannot be neglected.

- The state of the art in terms of testing and refactoring tools for analytics is far behind that of Agile software development tools (James Shore and Warden 2007).
- Integration testing for data analytics may be limited by the scale and complexity of the data. A complete run through of the analytics code can take hours of data processing. Unit tests are more difficult to define as analytics code is a series of data manipulations rather than compiled code.

Others have begun adapting Agile software development to data analytics (Collier 2011; Sunna and Agrawal 2010) when working in the context of data warehouse development. However, the unique challenges of guerrilla analytics make a traditional Agile implementation challenging due to time constraints, existing skill sets, and the limited time for up-skilling within the projects. Within this work we look to identify a minimum set of lightweight agile-inspired principles for guerrilla analytics.

Research Methodology

This work follows Design Science principles by (Hevner et al. 2004) within a design process that facilitates the engagement of scholars as well practitioners. The approach is closely aligned with the three design science research cycles proposed by Hevner: Relevance Cycle, Rigor Cycle and Design Cycle (Hevner 2007). To reach understanding of how a data analytics team acts and needs to react in guerrilla analytics project scenarios we examined the key factors in successful guerrilla analytics projects. Case studies are considered to be a suitable research approach for this study since it is exploratory in nature, with the intention of investigating guerrilla analytics techniques in a real-life context (Yin 2009).

The initial principles were defined using a mode of argument called induction by simple enumeration that rationalizes that "if a generalization has been positively instantiated in a large number of cases, and negatively instantiated in none, this makes it more likely that the generalization has been positively instantiated in a wide variety of circumstances" (Cohen 1970), and hence validity of inferences is increased. For the initial set of principles, we examined case studies of 4 different guerilla analytics projects. The studies involved collecting data and examining techniques on the 4 projects summarized in Table 1

Project	**Description**	**Analytics Team**	**Project Team**	**Duration**
Financial remediation of government client	Multiple undocumented system extracts and refreshes, ad-hoc spreadsheet data sources, daily delivery deadlines to distributed teams, reporting to high level government stakeholders	10 data analysts - statisticians, mathematicians, database experts and software engineers	100 forensic accountants and technology consultants	Multiple guerrilla analytics phases over 1.5 years with client reporting every 3 months and daily delivery of analytics.
Fraud investigation of manufacturing client	Single system extract, distributed team. Data mining and user profiling to detect fraud event.	2 experienced data analysts	1 business consultant	3 weeks
Forensic investigation of a bankruptcy	Multiple system extracts and poor quality bank statement data. Reporting to government committee	6 data analysts, recent graduates to managers	3 forensic accountants	2 months
Risk remediation of bank mis-selling	Multiple system extracts, multiple ad hoc human keyed sources, daily delivery deadlines. Restricted to client systems	5 experienced data analysts	40 risk analysts	7 months

Table 1 Summary of case studies

The projects for participation in the case studies were chosen for a number of reasons:

1. Participation within Guerilla Analytics projects and an appreciation of the need for principles to manage was evident to the managers therefore there was a mutual motivation.
2. Projects had encountered successes but also significant issues and challenges. These contrasting experiences provide what (Yin 2009) calls a 'rich' and 'revelatory' case.
3. Willingness to participate and to invest time in participating in the research together with a supportive environment.
4. Project staff were well placed to provide valuable insights from the practitioner perspective. The authors were active participants within some of the projects.

For this research, we observed the challenges put before the analytics teams. The key-informant method was then used to understand the key challenges, successes achieved, and initiatives taken or planned (Bagozzi et al. 1991; Campbell 1955). The key informants were staff members of the project teams. These ranged from experts with several years of experience, to junior team members. We conducted face-to-face interviews (Yin 2009) with team members to elicit feedback on workflows, tools and team conventions that helped the teams respond to guerrilla analytics challenges. Respondents were free to convey their experiences and views, and expression of the socially complex contexts that underpin key success factors in guerrilla analytics projects. The interviews were conducted in a responsive (Rubin and Rubin 1995; Wengraf 2001) manner, allowing the researcher to follow up on insights uncovered mid-interview, and adjust the content and schedule of the interview accordingly. As principles emerged, follow-up interviews were arranged to elicit further, richer, more focused information. This was done to confirm, extend, and sharpen the evolving list of principles. The result of these studies was a set of principles for guerrilla analytics development.

Emerging Principles of Guerrilla Analytics Development

Data Manipulation

It is important to deploy a lightweight data manipulation workspace early in the project so that the team can commence work as soon as possible. The exact choice of the technology used can be heavily dependent on the available client systems.

Principle 1: Establish a Data Manipulation Environment (DME)

The most important component of a guerrilla analytics team's platform is its data manipulation environment. At its most basic, this is a shared space on a file server. Much valuable basic analytics can be achieved on the file system manipulating text files and spreadsheets. It quickly becomes necessary to bring a sophisticated data manipulation tool to bear on the analytics project. Data needs to be searched, joined, profiled and subset in a repeatable, scalable and efficient fashion. Such an environment is typically a Relational Database Management System (RDBMS).

Principle 2: Establish Common Data Manipulation Services such as Search and Cleaning

To avoid reinvention of the wheel, to promote consistency and to facilitate up skilling, a guerrilla analytics team needs a set of common data manipulation services. For example, since disparate unconnected datasets need to be brought together, some form of data searching is required to find common joins between sources. This usually takes the form of a lookup of keywords from fields in all relevant data tables. This approach allows data to be reconstructed from common keys and keywords that appear in front end reports and user interfaces that the client will be familiar with.

Many analytics search and cleaning tasks are made much easier and more succinct in code with the availability of regular expressions (Friedl 1997). These are a language for concisely expressing matching patterns for text. Fuzzy matching (Winkler 2006) allows text to be compared and their similarity reported using a variety of similarity metrics. This allows disparate data sources to be joined together where a join on key fields is not available. Fuzzy deduplication allows approximate duplicates across one or more data fields to be identified and measured.

Data Provenance

Effective data provenance is critical to ensuring the auditability of the analytics team and the deliverables they produce.

Principle 3: Separate key types of data

We can distinguish 3 fundamental types of data that must be isolated from one another:

- **System dumps:** this is raw data extracted from client *systems*. It typically consists of a set of tables from a client RDBMS or warehouse.
- **Ad hoc data:** this is raw data that is not part of a system extract. It could be a mark-up list from a business analyst working for the client or the analytics team. It could be a front end report from a system or a data mapping provided by the client.
- **Work products:** this is the data generated as part of the analytics team's work products. It covers both intermediate tables and the final result tables of a work product in the DME as well as presentations, documents etc. on the file system.

Separation avoids the risk of mixing data the analytics team has produced with the raw data that everything is built on.

Principle 4: Log and version control all data received and produced

The analytics team will very quickly begin to receive data from a variety of sources on a daily basis. It is critical that every piece of data received is logged as there is no way to know in advance which pieces of data will contribute to a critical deliverable that will be subject to audit. Logs should capture information such as the receipt date, the receiver in the analytics team, the person from whom data was received and the location in the data manipulation environment to which the data was uploaded.

All past data drops must be retained for auditability and reproducibility of historic work products. Version control of data means a convention for distinguishing multiple data drops from one another. The version of data should be easily determined from the location or naming of the data so there is minimal overhead on the team's workflow.

The analytics team will produce work products. These cover team activities such as presentations, meeting minutes, code and dataset outputs, control total checks etc. Every work product should be logged, capturing the analytics team member completing the work, the reviewer, the client or business analyst team member for whom the work was produced etc. As with data received, there is no way to tell in advance which work products will contribute to an audited deliverable or achieve some other level of criticality in the project. It is therefore imperative that all work products be tracked.

In an environment where data is being exchanged frequently, modified by humans and iterated, it becomes critically important to track every record received and every record produced. Row ordering in an output dataset is never guaranteed and users should have the freedom to re-sort and manipulate the analytics team outputs they receive. Problems then arise when users return data with questions or bugs. Hashing helps identify every unique row of data received and produced by the analytics team.

Principle 5: Version control all analytics data builds

As the project progresses it is important to prevent the reinvention of the wheel when it comes to common joins and patterns in the data. Work products will perform the same data cleaning, the same data joins and attempt to incorporate the same evolving sets of business rules. If it is not managed consistently it leads to a risk of inconsistency in team deliverables. The solution is to introduce versioned 'builds' of the data. These are analytical datasets that encapsulate team knowledge and facilitate teamwork products by making common views of the data available to all team members. Where understanding of the data is evolving, the knowledge encapsulated in builds will also evolve. Version controlling analytics builds ensures historical work products are reproducible against a version of team knowledge, as it existed at a given point in the project's past. As the project progresses and the team learns about the data, more of the

complexity and testing against key project reconciliation numbers will shift from ad-hoc work products to the build code. This simplifies ad hoc work and frees up the team to add value elsewhere.

Coding

Principle 6: Write end-to-end analytics code

Analytics development code is not software code. The exploratory and ad hoc nature of analytics coding often leads to code that must be run piece meal. An analyst typically codes a quick data profile or subset, tests a join, recodes or cleans a field etc. The resulting code is more a collection of code snippets rather than an end-to-end data processing script. The analyst and reviewer must know to select and run some subset of code segments while checking outputs. This is time consuming.

The solution is to write code that runs straight through without interruption. The data inputs and outputs of the code are clearly identifiable; the code tears down and rebuilds its environment at the start to avoid bugs due to old lingering datasets. Key datasets and numbers produced during the analytics process are written to tables and logs for easy review. This may seem simplistic to those more familiar with development against an established data model but our experience is that this principle is rarely followed in guerrilla analytics projects.

Many work products will involve the execution of more than one code file. This may be because of their complexity or indeed because the user must jump between various tools and languages at various stages of the production of the work product. A scripting language allows the analyst to glue together the various code files, running them all in the correct order. The scripting language may be simply the available operating system shell language or may be a scripting language with additional useful features like logging and testing frameworks. Whatever the implementation, automated code execution ensures that outputs are absolutely up to date and that there are no bugs due to lingering datasets from exploratory coding.

Principle 7: Design code and analytics for audit and reproducibility

Key deliverables will have to be fully documented and reconciled to one another and to key project numbers to satisfy audit requirements. If the above principles have been followed, then this should be a straightforward exercise. However we can go further by taking the following into consideration.

- **'Horizontal audit trail'**: always retain raw column names. Never modify raw column contents, instead modify a copy. Suffix cleaned and modified column names in a consistent manner.
- **'Vertical audit trail':** code that subsets a population of data should create filter fields initially rather than immediately removing records. Intermediate tables should be used for significant data steps. This allows an auditor to apply filters herself to step through key stages of the analytics logic.

Testing

A challenge for an analytics team exploring a poorly understood data source is to have some confidence around the correctness of the results it produces. By definition, this work has not been done before and so the challenge becomes one of how data inputs, implementation of rules and creation of work products can be checked for correctness.

Principle 8: Identify key project data indicators and test against them

There will often be key numbers that are known about the data. It may be a number of transactions, a number of customers, a value on a trial balance, the number of electoral regions in a news article etc. It is critical that work products are tested against these numbers where possible. This drives a change in the coding/design philosophy of deliverables. For example, if we need to provide transactions for a given month but only have known trial balance numbers for a financial year, we should construct our data to cover the full year, reconcile against the year (since that number is known) and only then use a simple subsetting to the month in question.

Principle 9: Check for consistency by testing against the team's previous outputs

There will be occasions where deliverables are modified and iterated through in collaboration with the end user. The team, under pressure and battling complexity, must not produce deliverables that are inconsistent with previous versions. Since all work products are logged and separated and data and builds are versioned, it is straightforward to build historical consistency checking into a work product.

Conclusions and Future Work

This paper introduced principles for implementing Guerilla Analytics Development in guerrilla analytics projects. By guerrilla analytics, we mean those projects where delivery timelines are extremely tight and changing, data is undocumented, varied and changing and there is often tight integration with non-analytics team members both as designers of outputs and generators of inputs. Outputs are subject to strict requirements of audit and reproducibility. We derived the initial principles using case studies of 4 forensic and financial remediation projects over 3 years. We are following several directions in our ongoing research.

- We continue to identify and validate Guerrilla Analytics development principles within projects with the intent to create a design theory.
- We aim to translate these principles into team capabilities and a maturity framework. By measuring their capabilities against a maturity framework, a team can assess how well it is progressing towards being a true guerrilla analytics development team.
- Fundamental to team capability is the skill set of its team members. We are enumerating the typical skills, toolsets and disciplines required in terms of creating a guerrilla data analytics team.

Disclaimer

The views expressed in this paper are the personal views of the authors Enda Ridge and Edward Curry and not those of KPMG and are not intended to be interpreted as such.

References

Bagozzi, R. P., Yi, Y., and Phillips, L. W. 1991. "Assessing Construct Validity in Organizational Research," Administrative Science Quarterly (36:3)Cornell University, Johnson Graduate School, pp. 421–458.

Campbell, D. T. 1955. "The Informant in Quantitative Research," The American Journal of Sociology (60:4)The University of Chicago Press, pp. 339–342.

Cohen, L. J. 1970. The implications of induction, Methuen.

Cohn, M. 2010. Succeeding with Agile: Software Development Using Scrum, Addison-Wesley Professional.

Cohn, M. 2011. Agile estimating and planning, Prentice Hall Professional Technical Reference.

Collier, K. W. 2011. Agile Analytics: A Value-Driven Approach to Business Intelligence and Data Warehousing, Addison-Wesley Professional.

Friedl, J. E. F. 1997. Mastering Regular Expressions, O' Reilly.

Hevner, A. R. 2007. "A Three Cycle View of Design Science Research," Scandinavian Journal of Information Systems (19:2), pp. 87–92.

Hevner, A. R., March, S. T., Park, J., and Ram, S. 2004. "Design Science in Information Systems Research," MIS Quarterly (28:1)MIS RESEARCH CENTER-SCHOOL OF MANAGEMENT, pp. 75–105.

James Shore, and Warden, S. 2007. The Art of Agile Development, O' Reilly, pp. 432.

Kohavi, R., Rothleder, N. J., and Simoudis, E. 2002. "Emerging Trends in Business Analytics," COMMUNICATIONS OF THE ACM.

Lavalle, S., Lesser, E., Shockley, R., Hopkins, M. S., and Kruschwitz, N. 2011. "Big Data, Analytics and the Path from Insights to Value," MIT Sloan Management Review (52:2)Massachusetts Institute of Technology, pp. 21–32.

Luhn, H. P. 1958. "A business intelligence system," IBM Journal of Research and Development (2:4)IBM, pp. 314–319.

Martin, R. C. 2003. Agile Software Development: Principles, Patterns, and Practices, Prentice Hall.
Rubin, H., and Rubin, I. 1995. Qualitative Interviewing: The Art of Hearing Data, The Modern Language Journal (Vol. 80)Sage, pp. 0.
Sunna, W., and Agrawal, P. 2010. "Enabling Agile BI with a Compressed Flat Files Architecture," Business Intelligence Journal (15:2).
Wengraf, T. 2001. Qualitative Research Interviewing: Semi-Structured, Biographical and Narrative Methods, Sage Publications Ltd.
Winkler, W. E. 2006. Overview of Record Linkage and Current Research Directions, U.S. Census Bureau, pp. 44.
Yin, R. K. 2009. Case Study Research: Design and Methods, Essential guide to qualitative methods in organizational research, (L. Bickman and D. J. Rog, eds.) (Vol. 5)Sage Publications, pp. 219.