Towards Automated Data Integration in Software Analytics

Silverio Martínez-Fernández, Petar Jovanovic, Xavier Franch, Andreas Jedlitschka
Fraunhofer IESE (Germany), UPC-BarcelonaTech (Spain)

12th Intl. Workshop on Real-Time Business Intelligence and Analytics (BIRTE@VLDB) 2018
Rio de Janeiro (Brazil), August 27th 2018
Software Analytics

Real-time Enterprise for software organizations → Software Analytics

Insights, Actions

Data sources

Collected runtime, development data

data analysis

data gathering
Data for software analytics: An example

- Data produced during:
  - process
  - system
  - usage

- Heterogeneous data sources:
  - sources carrying different information
  - storage in heterogeneous formats and tools
  - diversity versions, usage, environments, security options,...
Motivation and Agenda

- Data for software analytics come from **heterogeneous data sources**
  - Most of the current approaches analyze a limited amount of data sources
  - Integration becomes necessary to make data actionable for decision making!!

- An **ontology-based data integration**
  - capturing the semantics of relevant data for software analytics

- **Two approaches:**
  - Static \(\rightarrow\) integration by predefined views representing software quality factors
  - Dynamic \(\rightarrow\) integration with evolvable information requirements
An ontology for software analytics

- A class represents an entity in the software analytics domain
- The ontology is abstract for software development → technologies can differ to enable generalization
- Associations among classes (e.g., commits having the issues identifier in the commits)
Information requirements help to reason about software quality metrics coming from several classes, e.g.:

- Information requirement 1 (IR1). Analyze the last release of the software product, per module, ordered by changes, quality rule violations, code quality measures (e.g., complexity, comments, and duplications).
- Information requirement 2 (IR2). Examine the reliability of a release of the software product in terms of bugs found during testing and errors occurring at runtime, ordered by their resolution time.
Static Approach – Q-Rapids tool

Three steps:

1. Data gathering (gathering real-time data for the ontology classes)
2. Data integration and aggregation (software quality factors from multiple classes)
3. Alerts redirecting to raw data visualization for actionable analytics
Support for different data sources: **nine connectors**

- **Heterogeneous data**: providing valuable information about the process, system, usage

- **Scalability**: able to ingest huge amount of data per second (e.g., usage)
  - Initial infrastructure for Big data → attractive for companies to adopt
Static Approach. Step 2 - Data integration/aggregation

- Creation (and evolution) of a quality model for actionable analytics
- Aggregation of heterogeneous data sources into product factors → transparency

Objects in Kibana

- Creating a view including all the data relevant for the information requirement, example IR1.
- Portable while maintaining defined indexes generated by qr-connect (for the ontology classes)
- It helps decision making. Examples:
  - Prioritizing critical open bugs
  - Solving quality rule violations
Static Approach. Step 3 – Alerts and visualizations
Drawbacks of the static approach:

- Manual effort on the part of the designer to integrate, transform, and prepare data
- When information requirements evolve, the above process undergo several rounds

A system to automate the process of building data integration flows for the field of software analytics:

- Quarry starts from high-level information requirements (e.g., IR1 and IR2) and semi-automatically generates target data stores and a data flow to integrate data from different data sources and prepare them for further exploitation

IR2: Examine the reliability of a release of the software product in terms of bugs found during testing and errors occurring at runtime, ordered by their resolution time.

The user:
- Express the information requirement using Quarry graphical tool
- Express restrictions (e.g., Issue Type=Bug)
- Express aggregations (e.g., count(issues))
Quarry extracts the subset of the ontology concepts needed to answer the given requirement

Quarry generates the schema of the target store
Automatic integration of data from different sources is still a challenge for the software analytics domain.

An ontology-based data integration
- capturing the semantics of relevant data for software analytics

Two approaches:
- Static → integration by predefined views representing software quality factors
- Dynamic → integration with evolvable information requirements

The envisioned dynamic approach would reduce manual efforts
- Immediate future work is to conduct a case study to validate the benefits of the dynamic approach.
Questions