Impact Analysis of Relational Schema Changes on Native Language Queries

Andy Maule

Department of Computer Science, University College London

what's the problem?

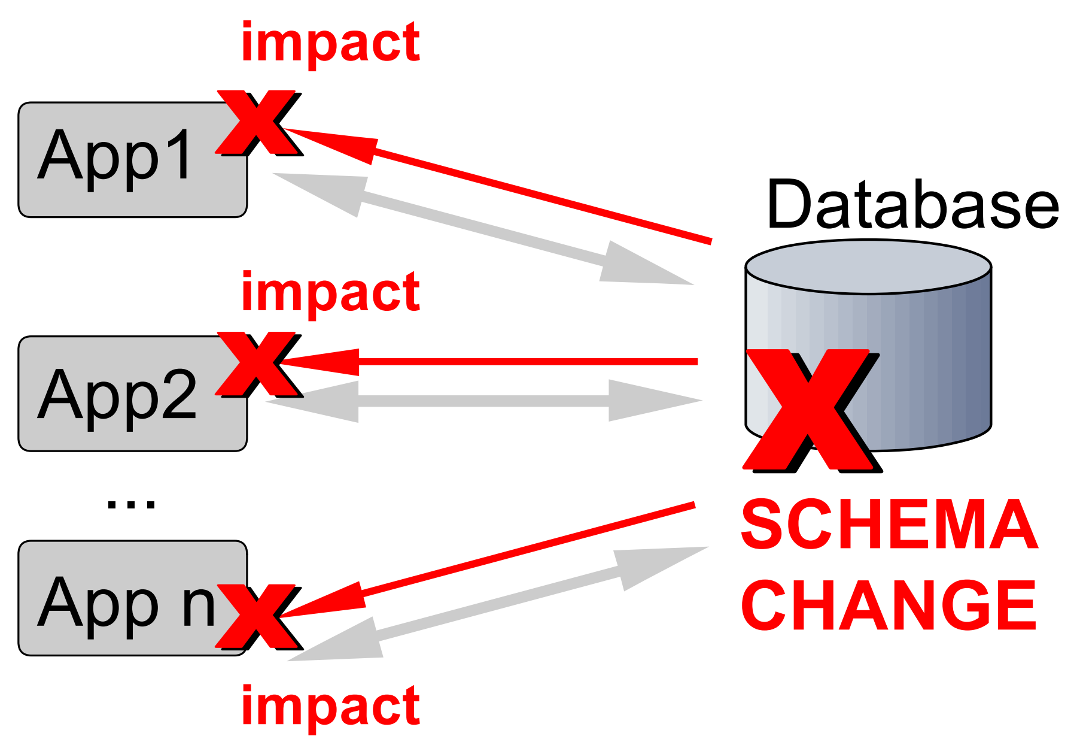
When a ***database schema*** the is changed, applications that use the database can be affected. Relational database schema changes are difficult to deal with because of lack of knowledge in two key parts of the change process, before and after a schema change. We propose an **impact analysis** technique to provide the required information.

*Before a schema change (**DBA**):*

- What will be affected if I make this change?
- How difficult to reconcile are these impacts?
- Are there any changes with less severe effects, that could be used instead?

*After a change (**application developer**):*

- What has been affected as a result of the change?
- How can I reconcile each impact?



what's the solution?

The proposed solution is split into 4 stages:

1. Create an abstract model by which we can specify dependecy relationships between artifacts.
2. Extract all necessary information from the applications and schema that is required to populate the model.
3. Perfom some reasoning on the instance of the model to predict the affects of a schema change.
4. Output the results of the calculation in a useful form suitable for providing all the information specified above.

Stages of analysis

1. Dependency model
2. Populate Model
3. Predict impact using model
4. Display results

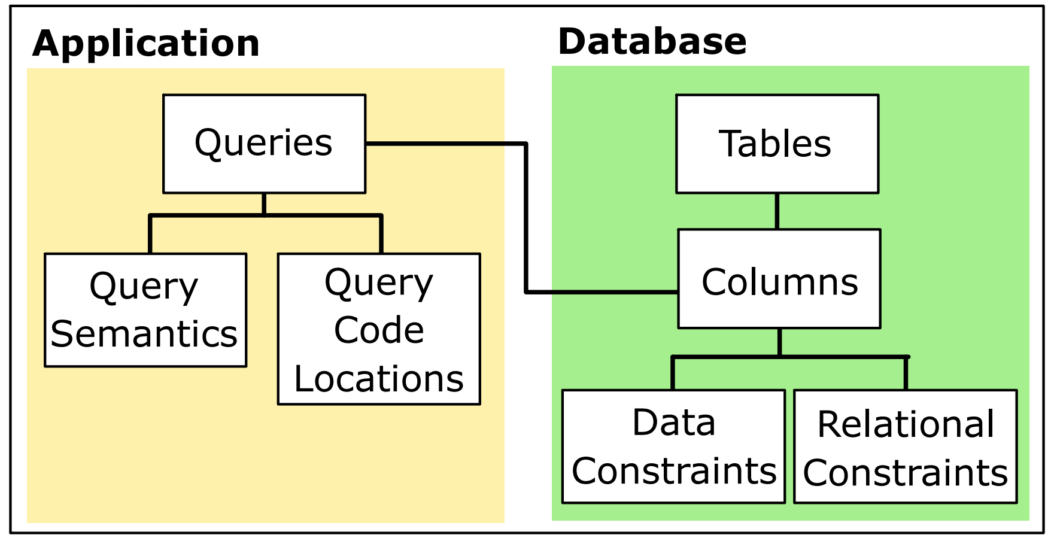
the dependency model

We create an abstract model to represent the system. We represent artifacts from both the database and the aplications.

Each artifact is related to other artifacts by dependency relationships that will help to determine the impact of schema changes upon the applications.

We formalise this model as **UML**, but we have displayed a simpllified version of the model here for clarity. We consider this model to be one possible example model, and not prescriptive. We expect to change and expand this model in future work.

Simplified abstract model



populating the model

To populate the model we need to extract the required data from the **schema** and the **applications**.

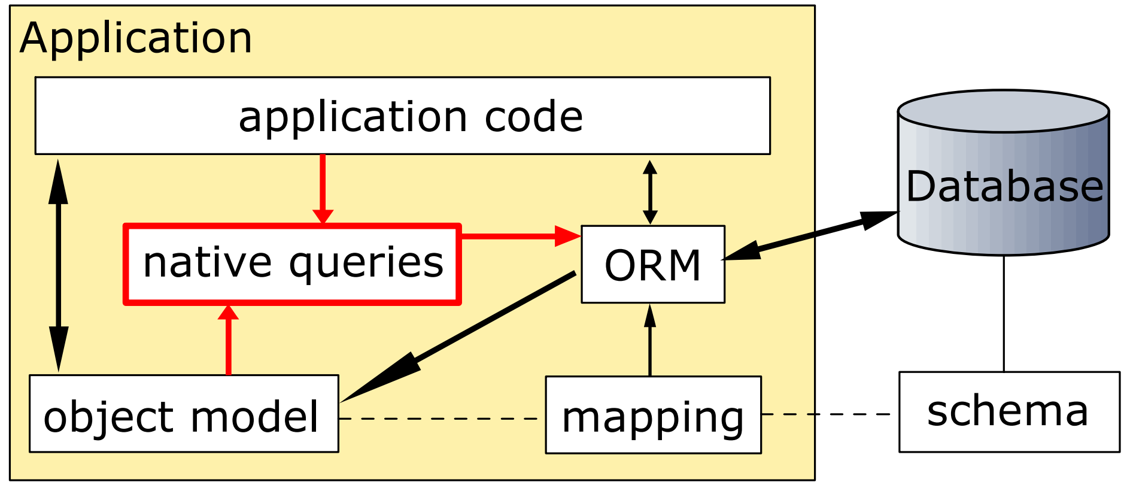
Extracting data from the schema is trivial. Extracting data from the application is very difficult.

We need to extract all possible queries from the applications, as well as the query semantics and code locations.

Queries are often specified as embedded strings, which are difficult to analyse. Recent advances in programming languages have introduced ways of specifying relational queries without using embedded strings or language extensions.

These statically typed queries are called **native queries** and they allow us to use standard **static program analysis** to extract query information.

Stages of analysis



predicting the impact

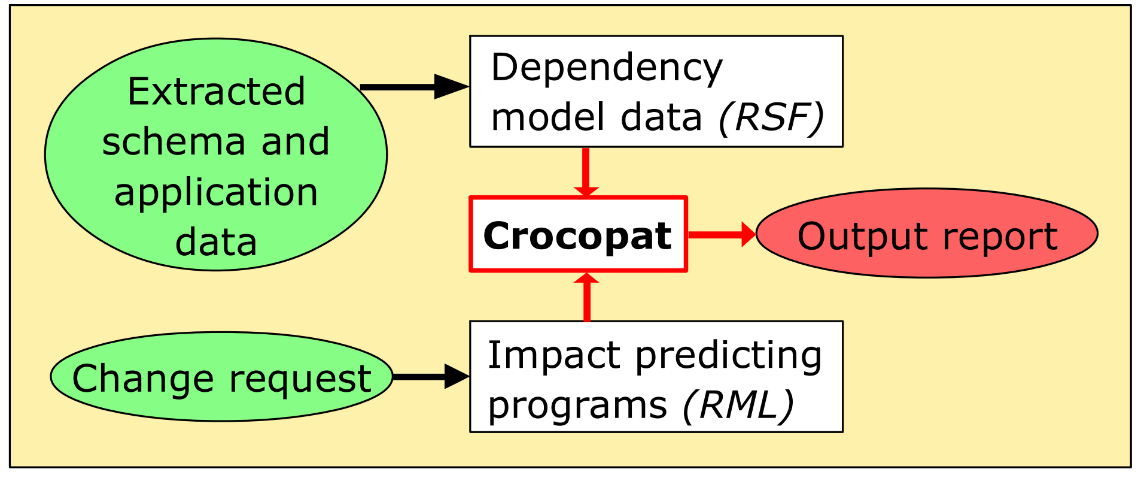
We use a tool called **CrocoPat** to efficiently make impact predictions.

We translate the **extracted data** into **RSF** format for use with CrocoPat.

For every change request we run an **RML** program. The RML program uses the dependency information from our model to predict what will be affected.

The output of this process is a report showing the predicted impact information.

Using CrocoPat


Extracted schema and application data
Dependency model data (RSF)
Crocopat
Output report
Change request
Impact predicting programs (RML)

conclusions and future work...

We have evaluated this approach on a real world case study. So far the technique looks promising, and appears to work well in applications that use native queries, however, more evaluation is required.

Future work...

We display the results as a simple textual report. Research is ongoing into how this data should be correctly displayed, possibly integrated with an IDE such as Eclipse.

We are also refining the data model and experimenting with different methods of extracting data from applications, including non native query applications.

The results of this research have been submitted for publication.

The submitted paper and more details about this project can be found at:

http://www.cs.ucl.ac.uk/staff/A.Maule

UCL

This project is funded in part by Microsoft Research Ltd.