The project's first task was an investigation of the current state-of-the-art in shared online databases, with the goal of recognizing their salient points and precisely identifying their shortcomings in dealing with data conflicts. The findings were the following: Shared online databases (Google Fusion Tables, QuickBase, Zoho Creator, Caspio Bridge, TrackVia and many more) enable online communities to collaboratively maintain their data. Unlike conventional databases, which were designed to provide only a back-end to enterprise applications, online databases offer a package consisting of a generic web application interface ready to be used by non-programmers and an underlying database. The project discovered that provision of a *generic* interface, *simplicity* of the interface and of the overall paradigm, and *pay-as-you-go support* are three salient features that distinguish online databases from conventional database systems.The genericity and simplicity of the interface are required to make the system suitable for non-experts. Additionally, the pay-as-you-go approach ensures that the system remains operable even before the users fully manipulate (integrate, clean etc) the data.

The project also discovered that shared online databases do not accomodate data conflicts, despite the fact that accomodation of data conflicts is easily found to be a requirement in many settings. In particular, the multiple users of the online community may hold conflicting beliefs, which they want to represent in the database. The reason for these conflicting beliefs could be different biases, different sources (whereas the one may be more up-to-date than the other), different interpretations of the same phenomena and many more. Instances of the latter often appear in the sciences, where researchers have contradicting opinions, about, for example, a genotype-phenotype map or a shadow on an X-ray. Each scientist or group often wants to record their opinion, even though conflicting beliefs, when entered into the database, will lead to conflicting data.

All current approaches towards data conflict management do not meet the three cornerstones of shared online databases, namely genericity, simplicity and resolve-as-you-go-support. particularly problematic is the failure of current approaches in addressing the genericity requirement: They force communities to agree on a single way to manage conflicts. Given the difficulty of agreement in a large-scale online community, the community's online database should ideally meet a fourth requirement: *Resolution personalization*, i.e., the ability of individual users (or groups of users) to resolve data conflicts in the way they see best.

The project's second task was to satisfy the above requirements, by developing the **Ricolla** (**R**esolve **I**nconsistencies in a **COLLA**borative environment) online database. Community members are able to both enter conflicting data and easily inspect the conflicts and resolve them. Moreover, members or subgroups can resolve the conflicts as they see best for their purposes, while being allowed to maintain different opinions with other individuals or subgroups. Finally, the conflict resolution happens in an "as-you-go" fashion, allowing members to use the system and query the data even before all conflicts have been resolved. In effect RICOLLA enables a community maintain a hugh number of "viewpoints" and query/analyze each viewpoint.

The project found a new tradeoff between simplicity, compactness and expressive power in data models aiming to capture data conflicts. The developed data model, called *ac-database*, and the corresponding generic report & update interface capture and represent data conflicts. The model's formal foundations draw from the database theory of possible worlds: an ac-database instance is a compact representation of a *set of possible worlds*. The literature contains several data models for representing sets of possible worlds (commonly referred to as data models for uncertain or incomplete data). However, Ricolla's data model differs from those in that it is tuned for the requirements of online database systems and is designed to give rise to an intuitive interface for such systems. In the interest of simplicity, it avoids the use of variables or complex provenance formulas in the data values representation, which, in contrast, are used in *c-tables*, *ULDBs* (used in the context of the Trio system) and *U-relations* (used in the context of the MayBMS system). The project showed that Ricolla's data representations are more compact than other data models, such as ULDBs. This, coupled with the direct visualization according to the data model, means lower cognitive load for users.

The project showed that the trade-off is effectively optimized in the sense described further down in this paragraph. In particular, the project quantified the expressive power penalty (that one pays in the interest of compactness and simplicity) by characterizing the *class of queries* whose answers (which are in general arbitrary sets of possible worlds) can be still represented in Ricolla's data model. In other words, the project identified the class of queries under which the data model is closed. By identifying it, it provides guarantees on which query answers can be represented in the data model without information loss (i.e., without losing any correlation between the tuples that might exist in the query answer). Furthermore, it shows that this is the largest class of queries that guarantees closure; i.e., the result of more expressive queries is not representable in a compact data model.

The project also developed a personalized conflict resolution policy, in the form of a set of **resolution actions** that allow community members to resolve individual conflicts. Different members can maintain different opinions by resolving conflicts in different ways. Another notable feature of the resolution actions is that they can be carried out not only on the base data but also on query answers. In the latter case, the system automatically translates the resolution actions on the query result to appropriate resolution actions on the base data that have the same effect. This allows community members to avoid resolving all conflicts and instead lazily resolve only those that affect queries in which they are interested.

Finally, by means of a comprehensive implementation and experimentation, the project showed that it is possible to maintain and query a sheer number of individual viewpoints, while avoiding exponentiality problems that prior solutions had when combining data of different viewpoints.