The subject of this dissertation is the modularisation of user interface code. We provide a conceptual solution which achieves the separation of the three user interface concerns we distinguish, namely the Presentation, Application and Connection concerns. As a proof-of-concept we have implemented DEUCE which uses a declarative programming language to specify the various user interface concerns. The declarative reasoning mechanism is used in combination with meta-programming techniques in order to compose the various UI concerns and the application into a final runtime application.

An important force within the domain of software engineering is the principle of separation of concerns. It is applied to modularise software systems such that various parts of the system can be dealt with in isolation of other parts. As modularisation decreases the amount of code tangling and code scattering, it increases the maintainability, evolvability and reusability of the various parts. Also within the domain of user interfaces this principle of separation of concerns can be applied for modularising user interface code and application code. However, the principle of separation of concerns has not been applied to a full extent to user interfaces yet. The developers are responsible for implementing both the application part and the user interface part, and the link between both parts. Furthermore, with the rise of new software challenges such as agile development environments and context-sensitive systems, these software systems and their implementation need to exhibit an increasing degree of variability and flexibility. Developing them still requires the developer to deal with tangled and scattered user interface code. Hence, evolving and maintaining user interface code and its underlying application has become a cumbersome task for the developer. This task can be alleviated by supporting the developers to apply the principle of separation of concerns to user interface code in order to improve the evolvability and reusability of user interface and application code.

While there exist contemporary techniques that are geared towards solving the problem of modularising user interface code and application code, these typically fall short in several ways. For example, the model-view-controller pattern which is
still a major player when it comes to separating user interface code from application code, does not tackle the problem of code tangling and scattering to a full extent. Other approaches such as Model-Based User Interface Development and User Interface Description Languages focus on a generative approach and are typically used for software systems where all interactions with the application originate from within the user interface only.

The solution we propose deals with separating the user interface code from its underlying application code for software systems in which application and user interface interact in both ways, for which several views can exist at the same time and in which dynamic user interface changes or updates are necessary. We elaborate on a conceptual solution to separate the following three user interface concerns. The presentation concern is related to the user interface itself and represents what the interface looks like and how it behaves. The application concern specifies how the application is triggered from within the user interface, and vice versa. The connection concern expresses how the presentation concern and application concern interact with each other and creates the link between both parts. We also postulate five requirements that are crucial for any solution that is aimed at a systematic modularisation of user interface code.

As a proof-of-concept implementation, we provide DEUCE (DEclarative User interface Concerns Extrication). This proof-of-concept uses a declarative meta-language (SOUL) on top of an object-oriented language (Smalltalk) and by doing so it provides a specification language to describe the entire structure and behaviour of the user-interface as well as its link with the application. This specification of the user interface concerns and the underlying application code are automatically composed into an final application for which a dynamic link with the original UI specification is maintained. DEUCE is put to practice by refactoring a Smalltalk personal finance application in order to validate that the conceptual solution does achieve an improved modularisation of user interface code. This modularisation removes code scattering and makes code tangling explicit in one location. Hence, the conceptual solution proposed in this dissertation establishes the separation of user interface concerns from a programmer’s perspective.