Towards Patterns for Defining and Changing Data Collection Instruments in Mobile Healthcare Scenarios

Johannes Schobel, Rüdiger Pryss, Marc Schickler, Manfred Reichert
Institute of Databases and Information Systems, Ulm University, Ulm, Germany
{*johannes.schobel, ruediger.pryss, marc.schickler, manfred.reichert*}*@uni-ulm.de*

***Abstract*—Especially in healthcare scenarios and clinical trials, a large amount of data needs to be collected in a rather short time. In this context, smart mobile devices can be a feasible instrument to foster data collection scenarios. To enable domain experts to create and maintain mobile data collection applications themselves, the QuestionSys framework relies on a model-driven approach to digitize paper-based questionnaires. This digital transformation is based on manual as well as automated tasks. The manual tasks applied by the domain experts can be eased by the use of change patterns. They describe features to easily add or delete the elements of a questionnaire. This work summarizes crucial change patterns and shows how they can be applied in practice. We believe that the patterns constitute an important means to implement sophisticated mobile data collection applications by domain experts themselves.**

***Keywords*-Mobile data collection, process-driven data collection, electronic questionnaires, change patterns.**

I. Introduction

In various application domains, *paper-based questionnaires* (e.g., instruments) are frequently used to collect large amounts of data. Recently, many hard-coded mobile applications were introduced, which enable data collection with smart mobile devices. These applications, in turn, increased the efficacy of the data collection procedure on one hand. On the other, frequent changes to these applications have revealed to be challenging. A more flexible support is provided by the QuestionSys framework, enabling domain experts to create data collection instruments on their own [1]. To achieve this goal, a sophisticated configurator component, combining end-user programming techniques with process management technology, was introduced [2].

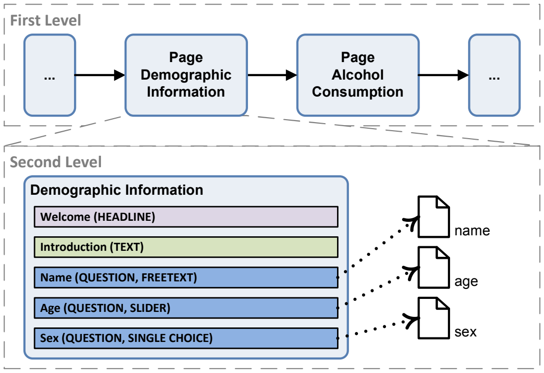


Figure 1. Different Levels of Change Patterns

In order to better understand the digital transformation of traditional paper-based questionnaires to complex *mobile data collection instruments*, structured interviews with 10 experts from various domains were conducted. In this context, more than 40 instruments from healthcare, psychology, logistics, and education were evaluated. Finally, a set of *change patterns* could be identified that shall enable domain experts to define or change data collection instruments.

II. Change Patterns

Table I illustrates the change patterns, which have been identified in the realized mobile data collection applications.[1] The patterns can be assigned to different levels. Each level, in turn, reflects different aspects of the data collection instrument (cf. Fig. 1). Patterns of the first level solely correspond to the structure (e.g., the flow) of the instrument, whereas change patterns of the second level refer to the content of questionnaire pages.

Structural Change Patterns (S) provide features to create and maintain the logic of a data collection instrument. For example, *pages* may be added or previously defined *instruments* may be inserted.

Content Change Patterns (C) enable the management of *elements* (e.g., headlines or questions) of a specific

	Tinnitus Research	Risk Factors during Pregnancy	Adverse Childhood Experiences	PTSD in War Regions	Learning Deficits among Students	Supporting Children after Accidents
S1 Insert Page	>5	>25	>80	>150	>10	>5
S2 Insert Block	0	>15	>80	>100	>15	>5
S3 Insert Empty Path	0	>15	>90	>250	>20	>5
S4 Embed Instrument	0	1	1	>20	0	0
C1 Insert Element	>60	>120	>200	>1500	>100	>40
C2 Move Element	>10	>40	>50	>500	>20	>10
R1 Cut Page	0	>5	>10	>30	0	0
R2 Merge Page	0	>5	>10	>15	0	0
R3 Move Page	0	>10	>15	>60	0	0
R4 Update Decision	0	>5	>30	>80	>15	0

Table I
Identified Change Patterns

[1] Estimated values, the projects are still ongoing.

page. Data elements for capturing answers are automatically created when using these patterns.

Refactoring Change Patterns (R) allow modelers to adapt an instrument to new requirements without violating validity constraints. For example, a page containing demographic questions may be moved, while other pages referring to these questions are updated accordingly.

The combined use of the identified change patterns allows domain experts to define or change mobile data collection instruments in a more flexible manner. Furthermore, they foster the continuous development and evolution of instruments (e.g., making use of the refactoring patterns). Fig. 2 illustrates how a set of change patterns may change an existing mobile data collection application.

In general, each identified pattern is described using a visual representation with an example from the considered real-world scenarios. Moreover, pre- and post-conditions for applying the patterns are listed (cf. Table II).

III. Related Work

The use of patterns in the context of reoccurring tasks is promising in various fields of computer science. [3] suggests a set of refactoring patterns for code fragments, while preserving the original behaviour of the source code. Furthermore, [4] proposes more than 20 patterns (e.g., *Observer* pattern). [5] proposes patterns for the communication between services. In the context of business process management, [6] introduces common process change patterns. The latter allow for comparing the expressiveness of respective information systems capable of executing process models. In [7], cloud architectural patterns for developing scalable applications are presented. These pattern allow for an easy installation and management on various environments.

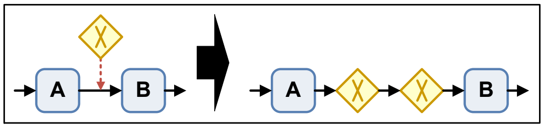
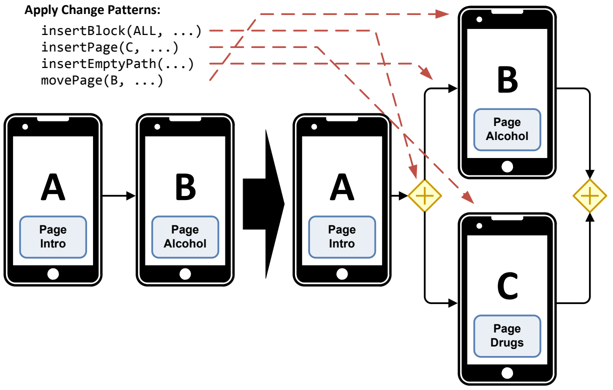
Name	Insert Block. Add a new block to an existing instrument. Available `types` are `IF`, `ALL`, and `REPEAT`.
Signature	`insertBlock(type, before, after)`
Example	Depending on the `type` of the block, various scenarios are possible: • `IF` blocks solely select *one path* based on already given answers during run time. • `ALL` blocks select *all paths* to be executed, however, the person interacting with the smart mobile device may choose its order of execution. • `REPEAT` blocks allow for repeating the content of the block multiple times. The amount of repetitions may be determined at run time (e.g., based on given answers) or are pre-defined by the domain expert (e.g., n times). 
Pre-Condition	The position to insert the `block` must be exactly specified; i.e., `after` must directly follow `before`.
Post-Condition	An empty block comprising a `split` and `join` gateway that are directly connected is inserted; For `IF` and `REPEAT` blocks, data elements for evaluating the conditions need to be connected using `READ` data edges.

Table II
STRUCTURAL CHANGE PATTERN S2: INSERT BLOCK



Apply Change Patterns:

```
insertBlock(ALL, ...)
insertPage(C, ...)
insertEmptyPath(...)
movePage(B, ...)
```

Figure 2. Applying Change Patterns to a Data Collection Instrument

IV. Summary

In the QuestionSys framework, automated and manual tasks are required to digitize paper-based questionnaires. Structured interviews identified a meaningful set of change patterns (i.e., manual tasks) needed in the various scenarios. The long-running projects from the healthcare domain revealed that the provision of sophisticated change patterns is indispensable. Providing change patterns in the context of mobile data collection applications is an important step to cope with the frequent application changes in practice on one hand. On the other, more longitudinal data can be gathered in a rather short project time. Altogether, the patterns allow domain experts without programming skills to better create and maintain instruments on their own.

References

[1] J. Schobel, R. Pryss, M. Schickler, M. Ruf-Leuschner, T. Elbert, and M. Reichert, "End-User Programming of Mobile Services: Empowering Domain Experts to Implement Mobile Data Collection Applications," in *IEEE 5th Int'l Conf on Mobile Services*, June 2016.

[2] J. Schobel, R. Pryss, M. Schickler, and M. Reichert, "A Configurator Component for End-User Defined Mobile Data Collection Processes," in *Demo Track of the 14th Int'l Conf on Service Oriented Computing*, October 2016.

[3] W. F. Opdyke, "Refactoring Object-Oriented Frameworks," Ph.D. dissertation, University of Illinois, 1992.

[4] E. Gamma, *Design Patterns: Elements of Reusable Object-Oriented Software*. Pearson Education India, 1995.

[5] A. Barros, M. Dumas, and A. H. ter Hofstede, "Service Interaction Patterns," in *Int'l Conf on Business Process Management*. Springer, 2005, pp. 302–318.

[6] S. Rinderle-Ma, M. Reichert, and B. Weber, "On the Formal Semantics of Change Patterns in Process-aware Information Systems," in *Proc. 27th Int'l Conf on Conceptual Modeling*, ser. LNCS, no. 5231. Springer, October 2008, pp. 279–293.

[7] C. Fehling, F. Leymann, J. Rütschlin, and D. Schumm, "Pattern-based Development and Management of Cloud Applications," *Future Internet*, vol. 4, no. 1, pp. 110–141, 2012.