FAKULTÄT FÜR !NFORMATIK
Faculty of Informatics

Diplomarbeitspräsentation

Extending the Peer Model with Composable Design Patterns

Master- /Diplomstudium:
Software Engineering & Internet Computing

Gerald Schermann

Technische Universität Wien
Institut für Computersprachen
Arbeitsbereich: Programmiersprachen und Übersetzer
Betreuerin: Ao.Univ.-Prof. Dr. Dipl.-Ing. eva Kühn

Context

The **Peer Model** is a programming model for modelling highly concurrent and distributed systems. It closes the gap between design and implementation by connecting the designer's view of clean and verifiable systems with the developer's aim of getting things done.

Reuse is an essential factor for software development. Developing each component of a new product from scratch is costly, includes risks and has negative influence on the time to market.

The aim of the software product line approach is to establish strategic reuse. A key element is software variability; it describes the software artifact's ability to be customized for use in a particular context. Properly used it leads to:

- ► increased product quality
- ► decreased time to market
- ► large-scale productivity gains
- ► increased consumer satisfaction

Patterns

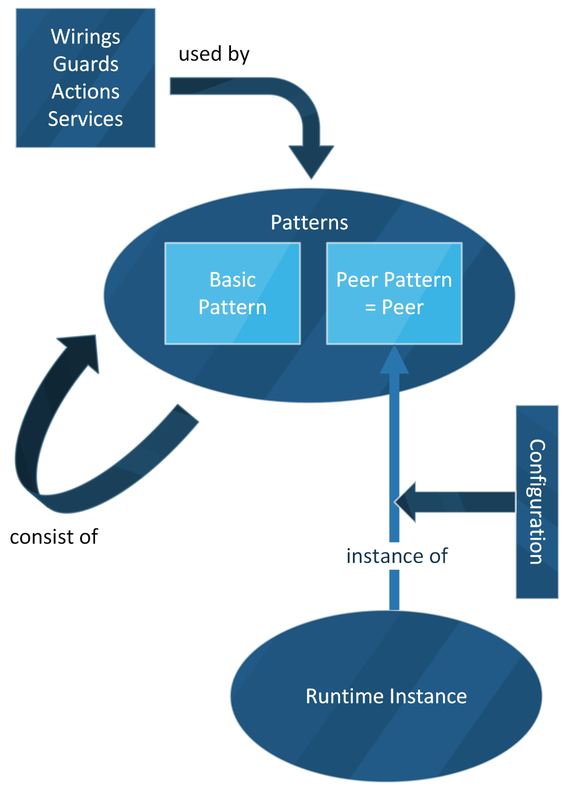
In the Peer Model, a pattern is a loose collection of components; these components are:

- ► Wirings
- ► Sub-patterns
- ► Services
- ► Guards/Actions

This concept introduces two pattern types:

- ► Basic Patterns and
- ► Peer Patterns.

A basic pattern is such a loose collection of components, a peer pattern is a special variant which is wrapped with input and output containers. Peer patterns can be treated like peers in models; they are self-contained components.



At design time users can rely on certain parameters which will be specified at configuration time. Such parameters represent the mentioned delayed decisions and introduce variability. They can be defined for:

- ► Location/Address information
- ► Operations
- ► Entry information
- ► Queries
- ► Services
- ► Guards/Actions

Contribution

Development of a pattern-based approach which provides such a software variability for the Peer Model. Design decisions are delayed and determined when concrete pattern instances are created. The concept allows

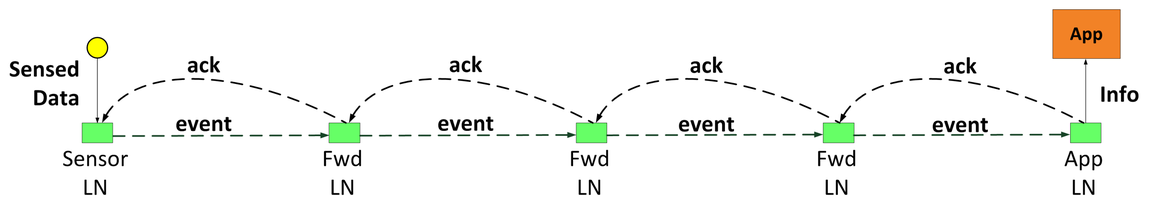
- ► defining generic patterns which depend on
- ► certain parameters specified at configuration time.

This variability allows for a far-reaching influence on the behaviour/ functionality of patterns and thus, it opens up the possibility for reuse on a large scale.

Moreover, systems designed with the Peer Model can be adapted to changing requirements without modifying the underlying architectural design and thus, cost- and time-intensive remodelling and refactoring work can be avoided.

Evaluation

The advantages of this approach were demonstrated by an example use case from the train traffic telematics domain, where signals of approaching trains are transferred from a sensor over multiple network nodes to a level crossing unit.

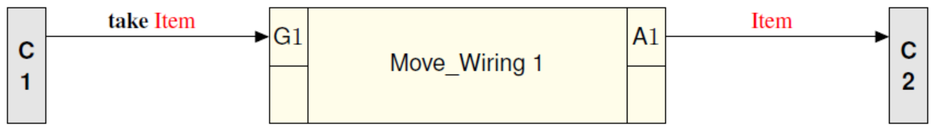


Various modelling concepts and tools were selected and with each of them, the use case was realized. In order to create a meaningful evaluation, a set of criteria was developed, emphasizing elements which are important for the design of highly distributed and concurrent systems and essential for the aspired-to variability.

Results

	CPN	Reo	Uppaal	BPMN	WS-BPEL	Actor Model	Peer Model
Composition	+	+	−	+	+	+	+
Reuse	+	+	+	+	+	+	+
Parametrization	∼	+	+	+	+	+	+
SoC	−	+	−	+	+	+	+
Dynamics	−	+	−	∼	+	+	+
Addressing	−	∼	+	∼	+	+	+
Scalability	−	−	+	+	+	+	+
Time	∼	∼	∼	∼	∼	∼	+
Toolchain & Docs	∼	∼	+	+	+	∼	∼
Simplicity	+	−	+	∼	−	+	+

Example: Move Pattern



The Move Pattern consists of a single wiring with one input link and one output link. It moves a specific 'Item' entry from a container C1 to a container C2.

- ► Item,
- ► C1, and
- ► C2 are specifiable.

Kontakt: gerald.schermann@gmx.at