FAKULTÄT FÜR !NFORMATIK
Faculty of Informatics

Diplomarbeitspräsentation

PeerSpace.NET
Implementing and Evaluating the Peer Model with Focus on API Usability

Masterstudium:
Software Engineering & Internet Computing

Dominik Rauch

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

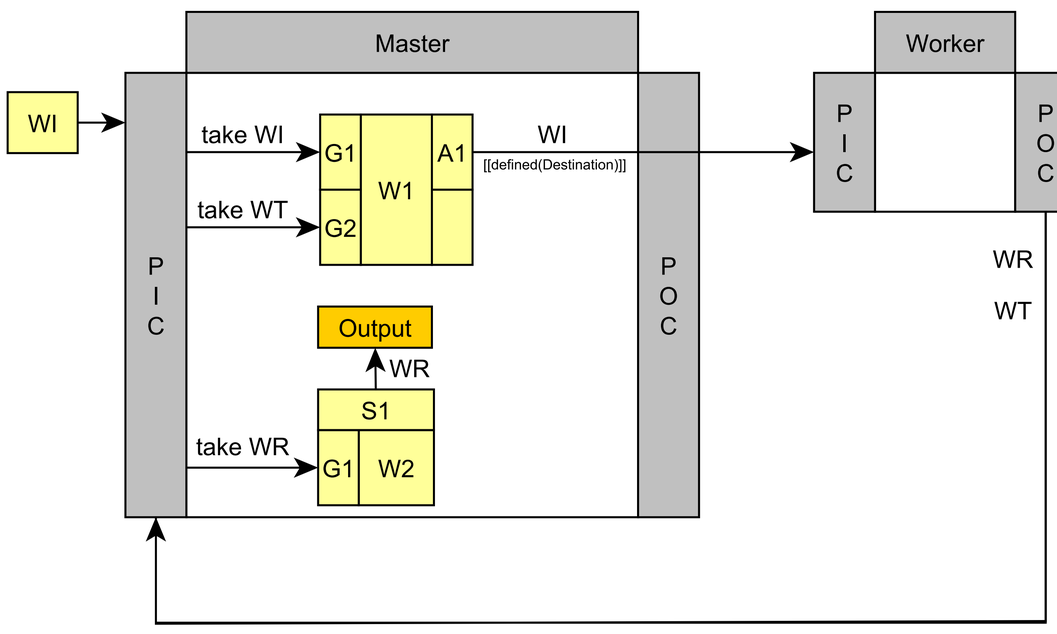
Introduction & Problem Statement

- Peer Model - a new modeling tool for distributed systems [1]
  - Enables scalable modeling
  - Allows composability and reuse
  - Separates coordination and business logic
  - Not restricted to certain patterns
- Practical establishment requires software implementation
  - A full-featured Peer Model implementation
  - Reusable distributed software components
  - Focus on extensibility and maintainability
- Earlier space-based frameworks faced "hard-to-use" criticism
  - Peer Model implementation shall have high API usability
  - Usage of modern API usability evaluation techniques

The PeerSpace.NET Framework

- The first Peer Model implementation based on .NET
- Allows to transform Peer Models into software components
- High developer comfort due to usability-focused API
- Advanced error handling possibilities
- Extensibility due to exchangeable implementation parts:
  - Container implementation
  - Communication layer
- Maintainability focus ensures future development
- Based on industry-proven Xcoordination AppSpace [2]

Example Peer Model



WI ... WorkItem

WT ... WorkerToken (marks a worker as "free")

WR ... WorkResult

Example PeerSpace Code

```
class MasterPeer : ApplicationPeerBase<DefaultPeerFactory>
{
    MasterPeer(PeerAddress address) : base(address) {
        Run();
    }

    void EmitNewWorkItem(WorkItem wi) {
        Peer.Emit(wi);
    }

    [Service] // Wiring W1
    void OnWorkItem(IServiceContext ctx, IEntry<WorkerToken> wt, WorkItem wi) {
        ctx.EmitWithDestination(wt.Origin, wi);
    }

    [Service] // Wiring W2
    void OnWorkItemResult(WorkItemResult wr) {
        Output(wr);
    }

    [CommunicationErrorCallback]
    void HandleCommunicationError(IPostContext ctx, Exception ex) {
        if (ctx.Attempt < 3) {
            // Try again in 5s
            ctx.RetryAfter(TimeSpan.FromSeconds(5));
        } else {
            // After three attempts put the work item back in the PIC
            var workItem = ctx.Entries.Single(e => e.Type == typeof (WorkItem));
            ctx.Peer.Emit(workItem);
        }
    }
}
```

Methodology

- State of the Art and Related Work analysis
- Gathering all framework requirements
  - Functional requirements
  - Non-functional requirements
- Implementing the PeerSpace Core component
- Designing & implementing the high-usability API
- Usability evaluation using qualitative & quantitative methods

Evaluation Results

- Comparison with well-known WCF framework
- Results from the cognitive dimensions framework [3]:
  - WCF not the best for complex coordination patterns
  - Even in SOA area the PeerSpace supersedes WCF usability
  - Penetrability & Expressiveness are main WCF pain points
- Results using automated API usability measurement:
  - Quantitative approach by Thomas Scheller [4]
  - Measures interface complexity
  - Very favorable results for PeerSpace API
  - More than 50% better scores
- In addition: less class coupling and less LOC
- Altogether: PeerSpace's API usability is much better than WCF's API usability

Download the PeerSpace framework at
https://www.nuget.org/packages/PeerSpace/

References

- [1] E. Kühn et. al., 2013
  Peer-based Programming Model for Coordination Patterns
  In *Coordination Models and Languages, p. 121-135. Springer*.
- [2] T. Scheller, 2010
  Xcoordination Application Space, `http://xcoappspace.codeplex.com/`
  Last accessed on 2014-04-18
- [3] S. Clarke & C. Becker, 2003
  Using the Cognitive Dimensions Framework to evaluate the usability of a class library
  In *Proceedings of the First Joint Conference of EASE PPIG (PPIG 15)*.
- [4] T. Scheller, 2014
  A Framework for the Automated Measurement of API Usability *(submitted for publication)*.

Kontakt: dominik.rauch@gmx.at